Simulating a Multi-Stage Screening Network: A Queueing Theory and Game Theory Application

Xiaowen Wang, Cen Song and Jun Zhuang

Abstract Simulation is widely used to study model for balancing congestion and security of a screening system. Security network is realistic and used in practice, but it is complex to analyze, especially when facing strategic applicants. To our best knowledge, no previous work has been done on a multi-stage security screening network using game theory and queueing theory. This research fills this gap by using simulation. For multi-stage screening, the method to determine the optimal screening probabilities in each stage is critical. Potential applicants may have access to information such as screening policy and other applicants' behaviors to adjust their application strategies. We use queueing theory and game theory to study the waiting time and the strategic interactions between the approver and the applicants. Arena simulation software is used to build the screening system with three major components: arrival process, screening process, and departure process. We use Matlab Graphic User Interface (GUI) to collect user inputs, then export data through Excel for Arena simulation, and finally export simulation from the results of the Arena to Matlab for analysis and visualization. This research provides some new insights to security screening problems.

This research was partially supported by the United States National Science Foundation (NSF) under award numbers 1200899 and 1334930. This research was also partially supported by the United States Department of Homeland Security (DHS) through the National Center for Risk and Economic Analysis of Terrorism Events (CREATE) under award number 2010-ST-061-RE0001. However, any opinions, findings, and conclusions or recommendations in this document are those of the authors and do not necessarily reflect views of the NSF, DHS, or CREATE. Cen Song and Jun Zhuang are the corresponding authors.

X. Wang · C. Song (✉) · J. Zhuang
Department of Industrial and System Engineering, University at Buffalo, Buffalo, USA
e-mail: censong@buffalo.edu; songcen22@gmail.com

X. Wang
e-mail: xwang54@buffalo.edu

J. Zhuang
e-mail: jzhuang@buffalo.edu

K. Hausken and J. Zhuang (eds.), *Game Theoretic Analysis of Congestion, Safety and Security*, Springer Series in Reliability Engineering,
DOI 10.1007/978-3-319-13009-5_3

Keywords Security screening policy · Two-stage queueing network · Waiting time · Game theory · Imperfect screening

1 Introduction

1.1 Background

Nowadays, security screening are very important in many fields, including airport security screening [14], visa application [7], and customs inspection [15]. Screening process can not be perfect, and there exist *type I* and *type II* errors [5]. *Type I* error is the incorrect rejection of a true null hypothesis. In a screening system, if a good applicant is rejected, the approver is said to have a *type I* error. By contrast, *Type II* error is the failure to reject a false null hypothesis. In a screening system [21], if a bad applicant is approved, the approver is said to have a *type II* error. Because of these errors, it adds costs to the approver. Moreover, these errors could pose serious threats to the public, such as security and safety issues. Therefore, multi-stage screening process is introduced to reduce errors.

After the 9/11/2001 attack, the U.S. government requires 100 % scanning of all U.S. bound containers by radiation detection and nonintrusive inspection equipment at a foreign port before getting them loaded on vessels [3]. The Transportation Security Administration developed the Certified Cargo Screening Program for explosives on a passenger aircraft to get 100 % screening [25]. It results in a longer waiting time for the passengers to pass the security system. Moreover, reducing the probability of screening, although it would lead to less waiting time, may fail to catch some bad applicants. For each stage, low screening probability causes more errors while high screening probability causes congestion. In a visa application for a particular tourism destination, if a good applicant knew before he applied, that the waiting time is long, he would not apply. It is a loss for the destination country, because of the potential loss of economic contribution. In order to deter bad applicants and to attract good applicants to the most, a balance between congestion and intensity of screening should be achieved. In an airport security screening process, such a long security check time may result in passengers missing the flight. The flight schedule for those who missed will be rescheduled. It will result in seats unoccupied in the missed flight, which is such a waste. Meanwhile, the flight to which passengers are rescheduled to might not have enough seats for them, because company usually over sell tickets to maximize their benefits [6]. The passengers who got their itinerary changed are causing unbalance flows among the airline schedules. Due to butterfly effect [28], it will lead to many more problems in the future. According to the data of 1980–2012 annual passenger number for Newark Liberty International Airport [18], there are huge amount of people arriving at the airport. To avoid heavy congestion as well as to deter adversaries, a proper screening probability is required to the security screening.

Since 1970s, researchers have studied security screening with queueing models [9]. We study the multiple stages of screening system and find out how to predict applicants' behavior by applying queueing models and game theory. Based on the simulation results, we find the optimal strategy for the approvers.

Game theory is a study of strategic decision-making [17]. Strategic interdependence is present in a social situation when what is best for someone depends on others' choices. The best strategies for attackers and defenders are analyzed by balancing protection from terrorism and natural disasters and by considering resource allocation [31]. The optimal inspection policies for security agency balance the inspection probability and average delay time and consider the adversary strategic gaming behavior [8]. The optimal proportional inspection using game theory is analyzed to achieve the most cost-effective combination of deterrence and detection [2]. In the model, we assume that the decision makers are rational. Each player maximizes his payoff, given his beliefs about the strategies used by other players [24]. In this screening system, game players include applicants (good and bad) and approvers, whose actions impact each other. Knowing other players' potential best responses, players make their optimal decision. We apply game theory to construct the dynamic system of screening system in this paper.

Queueing theory is the mathematical study of waiting lines [23]. In queueing theory, a model is constructed so that queue lengths and waiting time can be predicted [23]. Single $M/M/1$, multiple $M/M/c$, single channel and multiple stages, and multiple channels and multiple stages models are used to estimate the truck delay in the seaport [29]. The $M/M/N$ queuing models are developed to quantify truck congestion at primary scanning, and Monte-Carlo simulation is used to analyze the risk of containers missing vessels at secondary inspections [1]. An $M/M/m$ queuing model is designed and applied into an airport security system to analyze the optimal number of security gates [20]. A queuing network and discrete event simulation are used to test the effects of baggage volume and alarm rate at the security screening checkpoint [4].

In general, there are three ways to study the phenomena (fact or occurrence): analytic modeling [22], simulation [12] and experiment [11]. As phenomena becomes more and more complex, analytic models may prove to be overly simplified, and some complex models cannot have analytical solutions. Meanwhile, sometimes experiments are not able to be performed or are too expensive to conduct. Simulation provides a way to meet our needs for cheaper, faster, and more practical data [16]. Compared to various simulation methods, computer simulation might be the most widely used one. Computer simulation is numerical evaluation using software designed to imitate the systems operation characteristics [10]. Different softwares can be chosen according to the different characteristics of the system. Pendergraft et al. [19] simulate an airport passenger and luggage screening security screening system in a discrete event way. We need to simulate screening system, which consists of queues and decision tree. It can be simulated by entity flows such as items or passengers constrained by conditions. In simulation, queueing models are often used for rough cut and condition setting [30]. We use queueing models to set conditions, making the screening system more accurate.

GUI in Matlab [13] provides point-and-click control of software application. Applied with user-defined functions, Matlab GUI can fulfill the following tasks without users knowing any command lines: data import/output, data analysis and plotting. In this paper, we apply GUI in Matlab for data analysis.

The rest of this paper is structured as follows: A description of the model is presented in Sect. 2. Designing Arena to simulate this screening system is discussed in Sect. 3. Matlab GUI design is discussed in Sect. 4. A numerical experiment and data analysis are provided in Sect. 5. The conclusion and discussion on some possible future work and application are provided in Sect. 6.

2 The Model

Figure 1 shows the flowchart of the screening system. Potential applicants classified as good and bad applicants decide whether to enter this system or not. Once they enter, they may go through several imperfect screening stages based on the screening probabilities at each stage. If they are screened, they would enter an $M/M/1$ service queue, which follows a first-in first-out rule. Based on the imperfect screening results, the suspected bad applicants are denied, while others are further to be determined to be screened or passed the system. Some applicants who are not screened at all are defined as good and they can pass the system.

2.1 Notation

Table 1 lists the notation that is used throughout this paper. We define the candidates as those who have the intention to enter the system but not certain enough to be classified as *potential applicants*. Those who exactly enter the system are defined as *applicants*. Applicants are divided into two groups: *good applicants* and *bad applicants*. There is probability P that the potential applicants are good. The potential applicants enter the system with a Poisson arrival rate of Λ, including

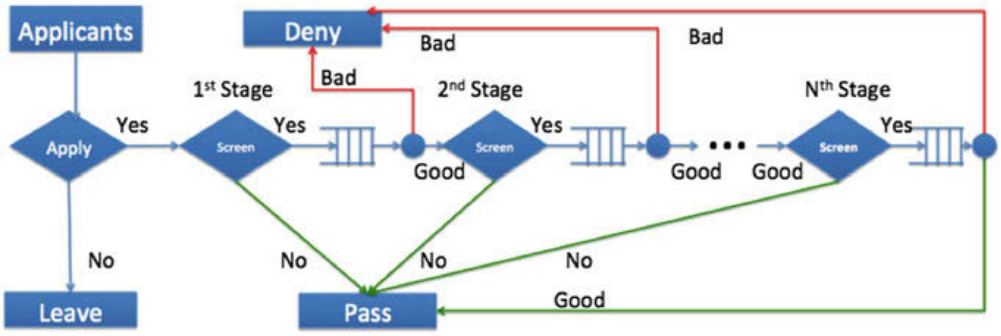


Fig. 1 Flowchart of screening system

Table 1 The notation used in this chapter

Λ	Potential applicants' total arrival rate
P	Percentage of good potential applicants in potential applicants
μ	Service rate in screening point
$i = 1, 2, 3, \ldots, N$	Stage number
Φ_1	Probability of screening in the first stage
Φ_{ig}	After passing the $(i-1)$th stage, probability of screening in ith stage
Φ_{ib}	After failed the $(i-1)$th stage, probability of screening in ith stage
α	Probability of reject good applicants
β	Probability of approving bad applicants
r_g	Reward of passing for good applicants
c_w	Waiting cost of good applicants
w	Total waiting time of good applicants
w_i	Waiting time in ith stage
r_b	Reward of passing for bad applicants
c_b	Cost of getting caught for bad applicants
R_g	Reward for approving good applicants for approver
R_b	Reward for denying bad applicants for approver
C_g	Cost for denying good applicants for approver
C_b	Cost for approving bad applicants for approver
N_{fb}	Simulation data: number of good denied
N_{fg}	Simulation data: number of bad approved
N_{rg}	Simulation data: number of good approved
N_{rb}	Simulation data: number of bad denied
U	Approver's utility
u_g	Expected utility for good applicants before deciding entering
u_b	Expected utility for bad applicants before deciding entering
P_{ag}	Calculated probability of passing for good applicants
P_{ab}	Calculated probability of passing for bad applicants
P_{db}	Calculated probability of getting caught for bad applicants
$p1$	Represents Φ_1 in simulation
$p2$	Represents Φ_{2b} in simulation
$p3$	Represents Φ_{2g} in simulation
pft	Represents P in simulation
tae	Represents α in simulation
tbe	Represents β in simulation
mu	Represents μ in simulation
lambda	Represents Λ in simulation
rewardg	Represents r_g in simulation
costw	Represents c_w in simulation
rewardb	Represents r_b in simulation
costb	Represents c_b in simulation

good potential applicants' arrival rate of Λ_g and bad potential applicants' arrival rate of Λ_b. The one who takes charge of the system is defined as *approver*. The approver screens the selected applicants with a service rate of μ.

To simplify the model, we assume service rates at each stage are equal. The *M*/*M*/1 queueing model is applied to study screening process at each stage. The probabilities of screening at each stage are defined as $\Phi_i, \Phi_{ig}, \Phi_{ib}$ for $i = 1, 2, \ldots, N$. From the second stage, suspected good applicants and suspected bad applicants would have different screening probabilities. For suspected good applicants we use Φ_{ig}, while Φ_{ib} is for suspected bad applicants. When the applicants enter the system, the approver screens them according to the probability Φ_{ig} or Φ_{ib}. At stage 1, those who are not screened will pass immediately. At stage $i(1 \leq i \leq N)$, those who are not screened will be approved or denied immediately according to the last stage screening results. Those who are screened are facing three consequences after one of the three stages of screening: approved, denied or enter to next stage $i + 1$. At stage N, those who are not screened will be approved or denied immediately according to the last stage result. Those who are screened at the last stage will pass or fail according to their own attributes (good or bad).

We assume that *type I* and *type II* errors are the same at each stage. We define *type I* error probability as α, and *type II* error probability as β. While there is a probability α that good applicants would be screened as suspected bad applicants at each stage, there is a probability β of vice versa.

We define the good applicants as those who receive reward r_g when getting approved, and pay a cost c_w per unit waiting time. Similarly, we define the bad applicants as those who receive reward r_b when getting approved, and pay a penalty c_b while getting caught. As we assume that $r_b \gg c_w$, waiting cost is neglected at this case. On the other hand, we define the approvers as those who receive reward R_g when approving good applicants and reward R_b when denying bad applicants. The approver pays cost C_g when denying good applicants and cost C_b when approving bad applicants. We collect data after simulating for the number of good approved N_{rg}, the number of good denied N_{fb}, the number of bad approved N_{fg} and the number of bad denied N_{rb}. The approver's utility is defined as U. We define the calculated probability of passing for good applicants as P_{ag} and the waiting time in queue as w.

2.2 Payoffs of Applicants and Approver

Approver's expected utility is shown in Eq. (1), where he maximizes his utility payoff.

$$U = N_{rg}R_g + N_{rb}R_b - N_{fg}C_g - N_{fb}C_b \tag{1}$$

For applicants, we also use their utilities to scale their payoffs. Good applicants' utility is defined as u_g in Eq. (2), where he maximizes his utility payoff.

$$u_g = P_{ag} r_g - w c_w \tag{2}$$

$$\begin{aligned} P_{ag} =& 1 - \left(\Phi_1 \sum_{j=2}^{n} \left(\alpha^j (1-\alpha)^{n-j} \left(\prod_{i=2}^{n} \Phi_{ig} \left(\prod_{p=2}^{j} \frac{\Phi_{pb}}{\Phi_{pg}} \right) \right) \right) \right. \\ & + \Phi_1 (1-\alpha)^{n-1} \alpha \left(\prod_{i=2}^{n} \Phi_{ig} \right) + \Phi_1 \alpha (1 - \Phi_{2b}) \\ & + \Phi_1 \sum_{j=3}^{n} \left(\sum_{i=2}^{j-1} (1-\alpha)^{j-i-1} \alpha^i \left(\left(\prod_{k=2}^{j-1} \Phi_{kg} \left(\prod_{p=2}^{i} \frac{\Phi_{pb}}{\Phi_{pg}} \right) \right) (1 - \Phi_{jb}) \right) \right) \\ & \left. + (1-\alpha)^{j-2} \alpha \left(\prod_{k=2}^{j-1} \Phi_{kg} \right) (1 - \Phi_{jb}) \right) \end{aligned} \tag{3}$$

To represent this series of problems, we use an *M*/*M*/1 queue as the queueing model, where there is a single server, unlimited waiting space, Poison arrival and exponential service time. Based on *M*/*M*/1 queue theory, at each screening point, we have waiting time W is shown in Eq. (4):

$$W = \frac{1}{\mu - \lambda} \tag{4}$$

For an N-stage screening, the total waiting time of good applicants is the summation of the screening waiting time at each stage, which are shown in Eq. (5).

$$\begin{aligned} w_1 &= \frac{1}{\mu - \Phi_1 \Lambda} \\ w_2 &= \frac{1-\alpha}{\mu - \Phi_1 \Phi_{2g} \Lambda} + \frac{\alpha}{\mu - \Phi_1 \Phi_{2b} \Lambda} \\ w_3 &= \frac{(1-\alpha)^2}{\mu - \Phi_1 \Phi_{2g} \Phi_{3g} \Lambda} + \frac{(1-\alpha)\alpha}{\mu - \Phi_1 \Phi_{2g} \Phi_{3b} \Lambda} + \frac{(1-\alpha)\alpha}{\mu - \Phi_1 \Phi_{2b} \Phi_{3g} \Lambda} + \frac{\alpha^2}{\mu - \Phi_1 \Phi_{2b} \Phi_{3b} \Lambda} \\ &\vdots \\ w_n &= \frac{(1-\alpha)^{n-1}}{\mu - \Phi_1 \Phi_{2g} \Phi_{3g} \Phi_{4g} \ldots \Phi_{ng} \Lambda} + \frac{(1-\alpha)^{n-2}\alpha}{\mu - \Phi_1 \Phi_{2b} \Phi_{3g} \Phi_{4g} \ldots \Phi_{ng} \Lambda} \\ &\quad + \frac{(1-\alpha)^{n-2}\alpha}{\mu - \Phi_1 \Phi_{2g} \Phi_{3b} \Phi_{4g} \ldots \Phi_{ng} \Lambda} + \cdots + \frac{(1-\alpha)^{n-2}\alpha}{\mu - \Phi_1 \Phi_{2g} \Phi_{3g} \Phi_{4g} \ldots \Phi_{nb} \Lambda} \\ &\quad + \frac{(1-\alpha)^{n-3}\alpha^2}{\mu - \Phi_1 \Phi_{2b} \Phi_{3b} \Phi_{4g} \ldots \Phi_{ng} \Lambda} + \cdots + \frac{\alpha^{n-1}}{\mu - \Phi_1 \Phi_{2b} \Phi_{3b} \Phi_{4b} \ldots \Phi_{nb} \Lambda} \\ w &= \sum_{i=1}^{n} w_i \end{aligned} \tag{5}$$

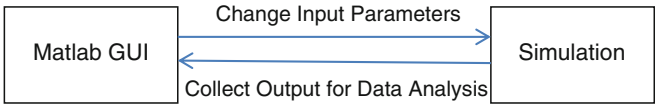


Fig. 2 Relationship between simulation and matlab GUI

Bad applicant's utility is defined as u_b is shown in Eq. (6), where he maximizes his utility payoff.

$$u_b = P_{ab} r_b - P_{db} c_b \tag{6}$$

We define the expected probability of passing for bad applicants as P_{ab} in Eq. (7), the expected probability of getting bad applicants caught as P_{db} in Eq. (8).

$$\begin{aligned} P_{ab} = 1 - \Bigg(& \Phi_1 \sum_{j=2}^{n} \left((1-\beta)^j \beta^{n-j} \left(\prod_{i=2}^{n} \Phi_{ig} \left(\prod_{p=2}^{j} \frac{\Phi_{pb}}{\Phi_{pg}} \right) \right) \right) \\ & + \Phi_1 \beta^{n-1} (1-\beta) \left(\prod_{i=2}^{n} \Phi_{ig} \right) + \Phi_1 (1-\beta)(1-\Phi_{2b}) \\ & + \Phi_1 \sum_{j=3}^{n} \left(\sum_{i=2}^{j-1} \beta^{j-i-1} (1-\beta)^i \left(\left(\prod_{k=2}^{j-1} \Phi_{kg} \left(\prod_{p=2}^{i} \frac{\Phi_{pb}}{\Phi_{pg}} \right) \right) (1-\Phi_{jb}) \right) \right) \\ & + \beta^{j-2} (1-\beta) \left(\prod_{k=2}^{j-1} \Phi_{kg} \right) (1-\Phi_{jb}) \Bigg) \end{aligned} \tag{7}$$

$$P_{db} = 1 - P_{ab} \tag{8}$$

By combining simulation and Matlab GUI, Fig. 2 shows the relationship between the tools we used.

3 Simulation

Arena is used to simulate the screening system. After each run, we get the numbers of applicants that are good but denied, bad but approved, good and approved and bad and denied. We define them as fake bad N_{fb}, fake good N_{fg}, real bad N_{rb} and real good N_{rg}. Figure 3 shows the whole structure of two-stage imperfect security screening process.

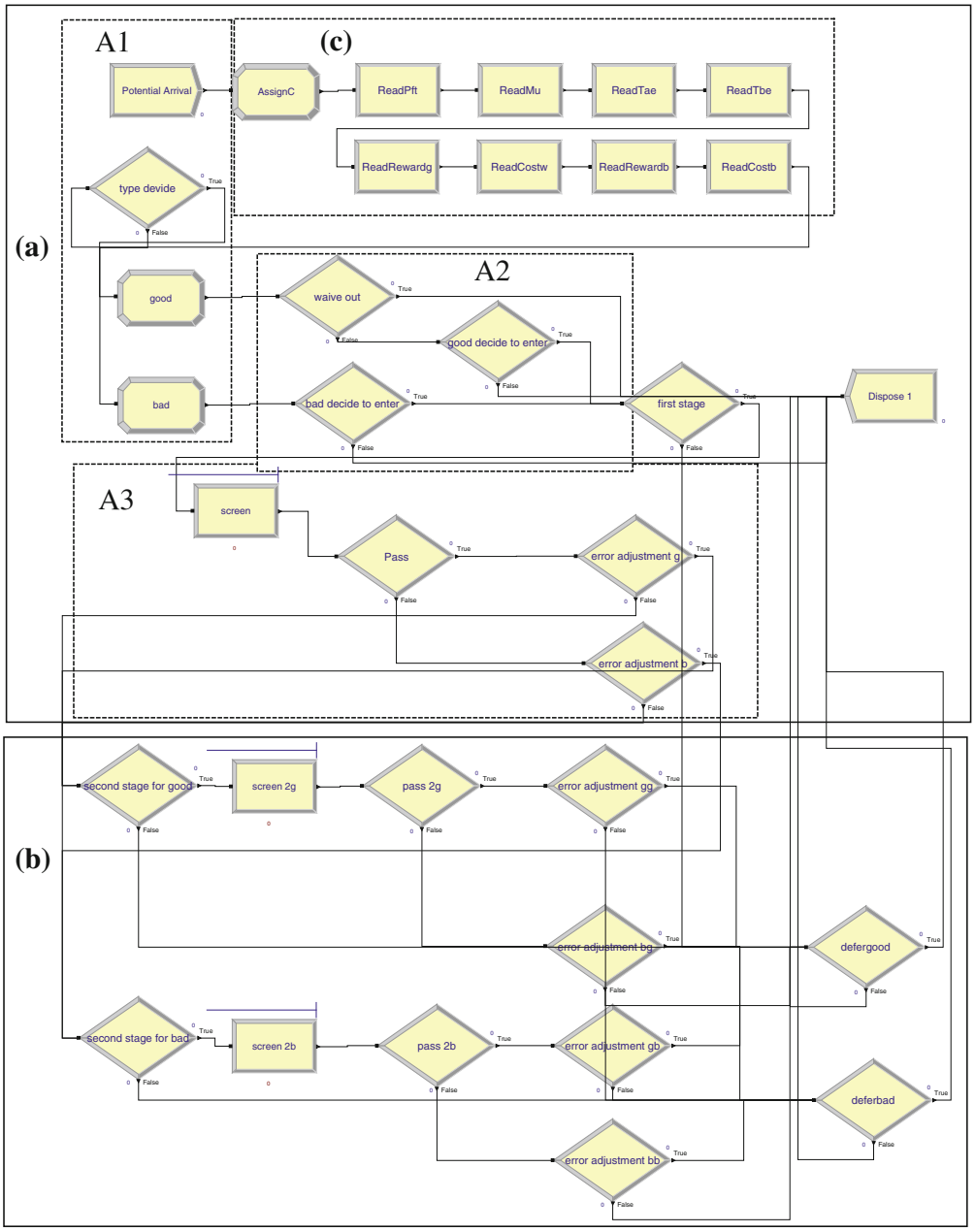


Fig. 3 Simulating two-stage imperfect security screening process

3.1 Simulating a Perfect One-Stage Screening System

Figure 3 part A excluding A2 shows the simulation of a one-stage imperfect screening process. First of all, we need to put a "Create" module named "Potential Arrival" to simulate potential applicants' arrival. It is Poison arrival as we described with the arrival rate of Λ, and "infinite" as max arrival. We define a variable "ar"

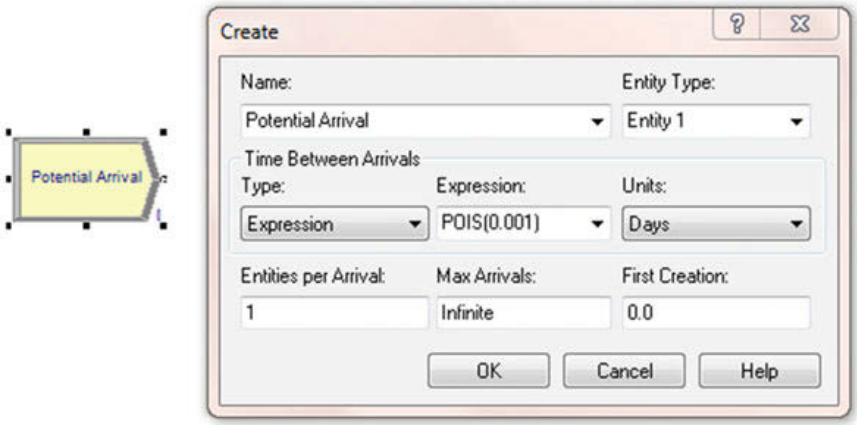


Fig. 4 Using "Create" module to simulate all applicants' arrivals

represent Λ, as Greek letters cannot be typed in Arena. "ar" has no value, and we assign a value to it in Sect. 5. Variable $\Lambda =$ "ar" per day holds the information that there is an average of "ar" persons arriving everyday. Figure 4 shows how to use "Create" module to simulate all applicants' arrivals. To set "Time Between Arrivals," there are several types to choose from: random[Expo], Schedule, Constant, and Expression. If you choose Expression, there are more Arena functions like WEIB and POIS you can choose from. In this case, we simulate a Poisson arrival process, then choose the "Type" of "Expression." The "Value Expression" box should be filled in with time value but not with rate value, "POIS(0.001)." The value of 0.001 is an approximate value, we may adjust it later in Sect. 5. Units should be defined correctly, so we put in "Days" here. In the last row, we define one entity at each arrival. "Max Arrival" is "Infinite" and first "Creation" is zero. The above simulates that every potential applicant follows a Poisson process, in an average of every 0.001 day to arrive.

Then, we divide potential applicants into two categories: good and bad. A "Decide" module named "type divide" can fulfill this task. We set it as "2-way" by chance, where chance is p, we use "pft" to present because p is a reserved variable in Arena. This module makes "pft" of potential applicants as good ones, followed by an "Assign" module named "good," which gives attribute "type" a value of 0. In some cases, entities in simulation need attributes to differentiate them from others. However, unlike many other program languages, these attributes can only be given values, but strings are prohibited. In this case, we give every entity an attribute named "type." Another "Assign" module named "bad" is added after the "False" output of "type divide" module, to assign attribute to bad applicants. If "type" equals zero, it means this entity represents good applicants. If "type" equals one, it means this entity represents bad applicants.

Figure 5 shows how to simulate two categories: good applicants and bad applicants. The variable "pft" can be found in "Variable" module, which is shown in Fig. 6. The "Initial Value" can be left blank for reading data from files in the

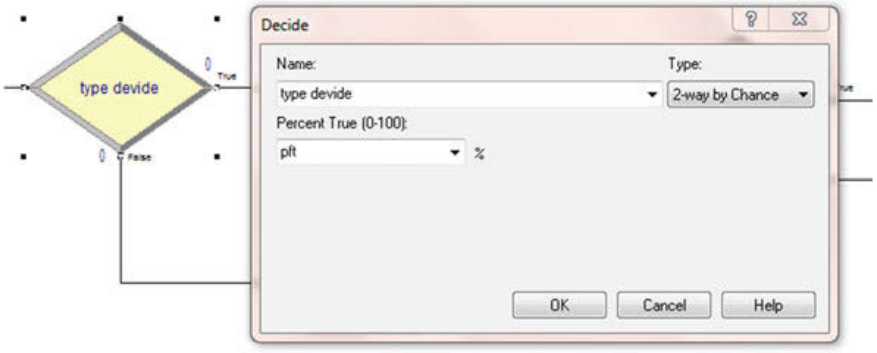


Fig. 5 Simulating arrivals of good and bad applicants

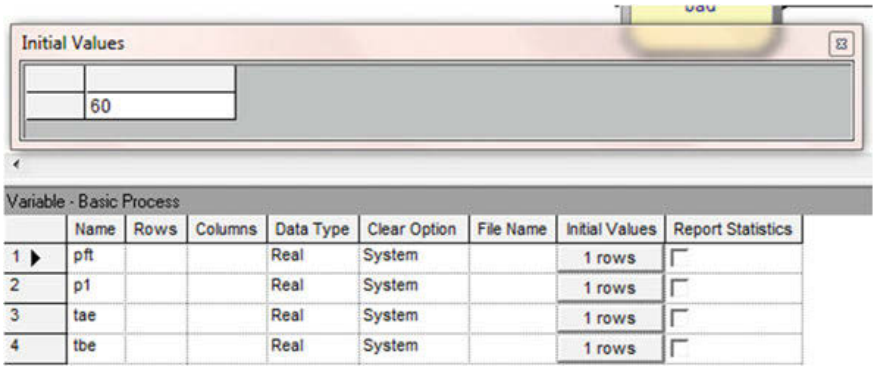
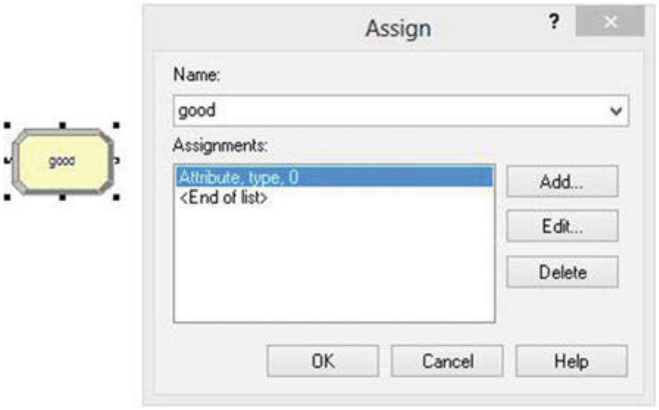


Fig. 6 Setting probability of good applicants as a variable p

Fig. 7 Assigning attributes to good applicants



future. We assign a value, for example, "pft = 60", which means that there is a probability of 60 % that a potential applicant is good. The numbers by the modules stand for the numbers of entity going through this route. Figure 7 takes good

applicants as an example, and shows how to assign attribute and differentiate the two groups of potential applicants.

In Fig. 3, part A1 simulates potential arriving applicants and divides them into two types. The key part of the system is screening. The model we discuss here is a two-stage screening model. We start from first-stage screening. It consists of two "Decide" modules and a "Process" module. One of the "Decide" module named "first stage" is to set as "2-way" by chance, whose chance is Φ_1. This module makes Φ_1 of applicants step into the screening process. Meanwhile, the rest of applicants would get an immediate pass. The "Process" module named "screen" is set the action as "Seize Delay Release," having "1" resource as the approver with a service rate μ (*Exponential Distribution*). The "Seize" represents the process of getting a free resource. An entity might have to wait in a queue until the resource is free. "Delay" represents the process time, and "Release" corresponds to the action of an entity releasing a resource, so that the next entity in queue could use the resource. This module represents the screening process, using *M*/*M*/1 queue. At the end of screening, the approver has to decide whether to approve or deny according to the type of the applicants (good or bad). The other "Decide" module named "pass" is put here, setting as "2-way" by condition, where condition is "type ≤ 0". We use the variable "p1" refers to Φ_1 to decide the screening chance to the applicants at the first stage.

In Fig. 3, part A2 shows the simulation of the potential applicants' decision-making. To simulate congestion cost, we assume potential applicants quitting causes cost. We design the condition nodes to simulate this. The condition nodes have the condition $u_g/u_b > 0$, where u_b and u_g represent the expected utilities for good applicants and bad applicants, respectively. There are two ways to simulate the decision: Static and Dynamic. In this paper, we simulate the static decision. There is no update information for the later potential applicants before entering. Assuming the potential applicants know the information about the screening system and use the information to make decision on entering the system, we apply queueing theory to obtain waiting time and probability knowledge to obtain utility. For the applicants, if utility is greater or equal to zero, they will enter the system. Only potential good applicants will pay waiting cost, because we assume $r_b \gg c_w$ for bad applicants. We have w, P_{ag}, P_{ab}, and P_{db} for two-stage screening system in Eqs. (9)–(12), respectively.

$$w = \frac{1}{\mu - \Phi_1 \Lambda} + \frac{1-\alpha}{\mu - \Phi_1 \Phi_{2g} \Lambda} + \frac{\alpha}{\mu - \Phi_1 \Phi_{2b} \Lambda} \tag{9}$$

$$P_{ag} = 1 - \left(\Phi_1 \Phi_{2g} \alpha (1-\alpha) + \Phi_1 \Phi_{2b} \alpha^2 + \Phi_1 (1-\Phi_{2b}) \alpha\right) \tag{10}$$

$$P_{ab} = 1 - \left(\Phi_1 \Phi_{2g} \beta (1-\beta) + \Phi_1 \Phi_{2b} (1-\beta)^2 + \Phi_1 (1-\Phi_{2b})(1-\beta)\right) \tag{11}$$

$$P_{db} = \Phi_1 \Phi_{2g} \beta (1-\beta) + \Phi_1 \Phi_{2b} (1-\beta)^2 + \Phi_1 (1-\Phi_{2b})(1-\beta) \tag{12}$$

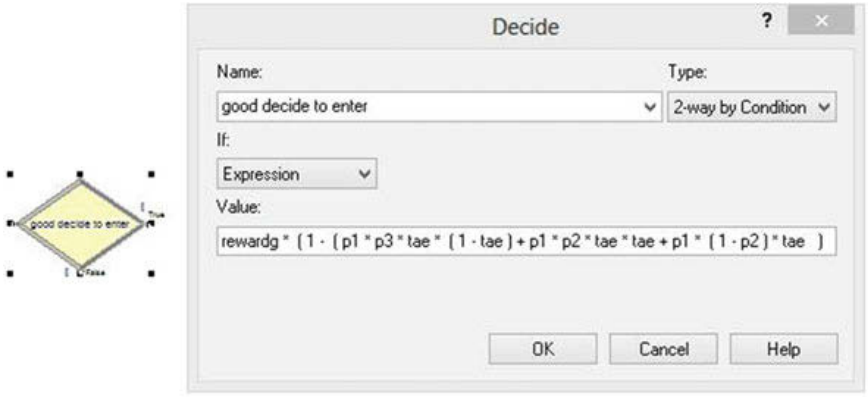


Fig. 8 Simulating entering condition for potential good applicants

Meanwhile, if $(\mu - \Lambda\Phi_1)$ or $\left(\mu - \Lambda\Phi_1\Phi_{2g}\right)$ or $(\mu - \Lambda\Phi_1\Phi_{2b})$ equals to zero, waiting time will be infinite. There is no point to entering the system, and it would eventually cause error in Arena. We add a new decision node to waive it out. Three "Decide" modules named as "good decide to enter", "bad decide to enter" and "waive out" are added before the screening process.

Figure 8 shows the simulation of the decision making for good applicants. For the flow of *good potential applicants*, they need to go through a "waive out," setting as "2-way by condition", where condition is $(\mu - \Lambda\Phi_1)$ or $\left(\mu - \Lambda\Phi_1\Phi_{2g}\right)$ or $(\mu - \Lambda\Phi_1\Phi_{2b})$ equals to zero. The values for μ and Λ can be put in according to Sect. 5. If the specific condition is 'yes,' the entity will leave the system, which means the good applicants will quit applying. If the specific condition is "no", the entity will go through "good decide to enter," setting as "2-way by condition," where condition is $u_g \geq 0$. If the specific condition is 'yes,' the entity will continue to stay in system, which means the good applicant will finally decide to apply. If the specific condition is "no", the entity will leave the system, which means the good applicant will quit applying. The two conditions in "waive out" and "good decide to enter," are set as "if" Expression. We need to use Expression Builder in Tools menu to formulate the "Value" of the "Expression." The "Value" of "Expression" for "waive out" is $mu - lambda * p1 == 0 || mu - lambda * p1 * p3 == 0 || mu - lambda * p1 * p2 == 0$. The "Value" of "Expression" for "good decide to enter" is written in the phrase of Arena as: $rewardg * (1 - (p1 * p3 * tae * (1 - tae) + p1 * p2 * tae * tae + p1 * (1 - p2) * tae)) \quad -costw * (1/(mu - lambda * p1) + (1 - tae)/(mu - p1 * p3 * lambda) + tae/(mu - p1 * p2 * lambda)) > = 0$.

Figure 9 shows the simulation of the decision-making for potential bad applicants. For the flow of bad potential applicants, they need to go through "bad decide to enter," setting as "2-way by condition," where condition is $u_b \geq 0$. If the specific condition is 'yes,' the entity will continue to stay in system, which means the bad applicants can apply. If the specific condition is 'no,' the entity will leave the system, which means the bad applicant would quit applying. The "Value" of "Expression" for "bad decide to enter" is written in the phrase of Arena as:

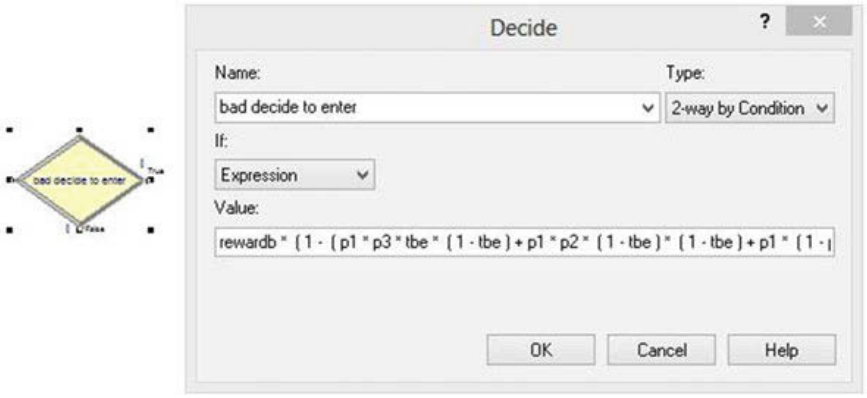


Fig. 9 Simulating entering condition for potential bad applicants

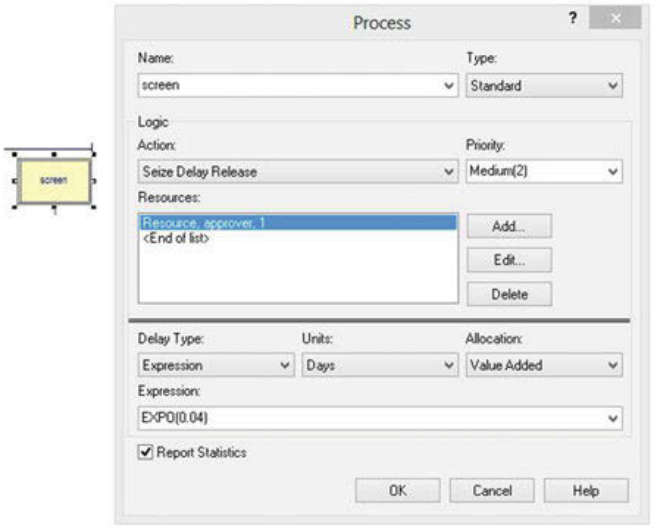


Fig. 10 Simulating the screening process

$rewardb * (1 - (p1 * p3 * tbe * (1 - tbe) + p1 * p2 * (1 - tbe) * (1 - tbe) + p1 * (1 - p2) * (1 - tbe))) - costb * (p1 * p3 * tbe * (1 - tbe) + p1 * p2 * (1 - tbe) * (1 - tbe) + p1 * (1 - p2) * (1 - tbe)) > = 0$. Those entities that leave the system will directly go to "Dispose" module.

Figure 10 simulates the screening process. In the "Logic" group, "Action" can be defined as Delay, Seize Delay, Seize Delay Release and Delay Release. We choose "Seize Delay Release". Then we add resource, which represents the server named as Approver, quantity = 1. "Delay Type" can be defined as Constant, Normal, Triangular, Uniform and Expression. As the server follows exponential distribution, we choose "Expression" and define the same unit as the arrival applicants: "Days." Allocation is Value Added. Expression is "EXPO(0.004)". The number 0.004 represents the service time in an average of every 0.004 day. We can adjust $\mu = \frac{1}{0.004}$ as in Sect. 5. Figure 11 simulates the approver that passes or fails applicants after screening. It depends on the attributes of applicants. The "Decide"

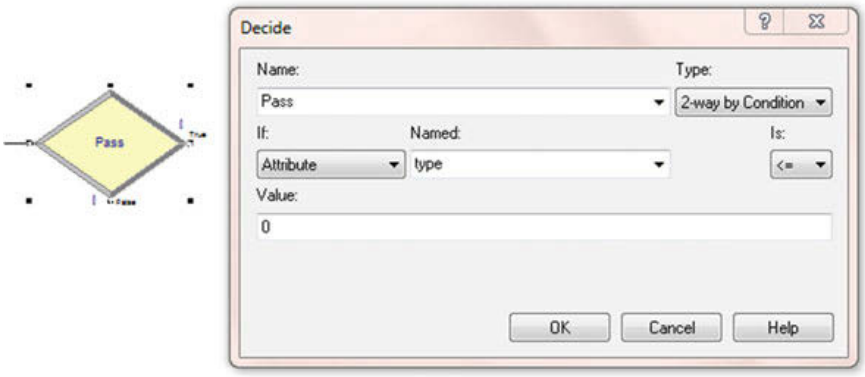


Fig. 11 Simulating the approver's decision on whether to pass or reject

module is defined as "2-way by Condition." If "Attribute: type ≤ 0, which is type = 0," then good applicants can pass. Otherwise, it is bad applicants, who are to be rejected. All the entity flows end up in "Dispose" module, which represents the completion of the process.

3.2 Simulating an Imperfect Multi-stage Screening System

In Fig. 3, part A shows the simulation of the first stage imperfect screening process. In Fig. 3, part A3 simulates the *type I* and *type II* errors. Since there exist *type I* and *type II* errors, we use two "Decide" modules to model and simulate them. After a former "Decide" module named "pass," the entities will be divided into two categories: good applicants "(type = 0)" and bad applicants "(type = 1)". Following the entity flow of good applicants, we put a "Decide" module named "error adjustment g", setting as "2-way" by chance, where chance is $1-\alpha$. It represents that the approver has *type I* error, leading to $100(1-\alpha)\,\%$ of good applicants as good and $100\alpha\,\%$ of good applicants as bad. Following an entity flow of bad applicants, we put a "Decide" module, setting as "2-way" by chance, where chance is $(1-\beta)$. It will represent the approver have *type II* error, leading to $100(1-\beta)\,\%$ of bad applicants as bad and $100\beta\,\%$ of bad applicants as good. We add two variables: "tae" to represent α and "tbe" to represent β. Good applicant group and bad applicant group have been updated to new *good applicants* group (good and bad), and new *bad applicant* group (good and bad).

To eliminate error, we apply multi-stage screening. In this simulation, we carry out two-stage screening to analyze. The two new groups of applicants step into second-stage screening. Both of them have the same scenario of modules as at the first-stage screening, expect that the screening probability is Φ_{2g} (for new *good ones*)/Φ_{2b} (for new *bad ones*) instead of Φ_1. In Fig. 3, part B shows the simulation of the second-stage imperfect screening.

We use "p2" and "p3" to represent Φ_{2g} and Φ_{2b}, respectively. "Decide" Modules "second stage for good," "screen 2g," "pass 2g," "error adjustment gg" and "error adjustment bg" simulate second-stage screening process for good ones. Modules "second stage for bad," "screen 2b", "pass 2b," "error adjustment gb" and "error adjustment bb" simulate second-stage screening process for bad ones.

3.3 Designing Input and Output Functions in Arena

In this simulation, the inputs include service rate (μ), percentage of potential good applicants in total potential applicants (*pft*), screening probabilities ($p1$, $p2$ and $p3$), *type I and type II* error (*tae* and *tbe*), rewards and costs for potential good applicants and bad applicants (*rewardg*, *costw*, *rewardb*, and *costb*). The outputs are numbers (N_{fb}, N_{fg}, N_{rg} and N_{rb}). For the convenience of data analysis in Matlab, we add output for recording current screening probabilities, service and arrival rates.

Due to the large amount of input and output data, several "ReadWrite" modules and "Record" modules are added. "ReadWrite" module can be considered as a bridge between *Arena* and *Microsoft Excel*. For each Excel file read to Arena, it is called "Arena File Name". "ReadWrite" module is not read directly from Excel file name, but from "Arena File Name." In "File" module, there is a table of all the Excel File names that particular "Arena File Name." For input data, the type of module is set as "Read from File". The action to "Assignments" is put in each settings of the "ReadWrite" module. Data that read from this module assigns value to the "Assignments". Therefore, there are eleven "ReadWrite" modules for input. We name them as "ReadPft", "ReadMu", "ReadP1," "ReadP2," "ReadP3," "ReadTae," "ReadTbe," "ReadRewardg," "ReadCostw," "ReadRewardb," and "ReadCostb." Figure 12 shows how to set the modules for input. Figures 13 and 14 list the modules and files in the simulation. In Fig. 3, part C shows how to read the input.

Arena can save data in a .csv file, which can be opened in Excel. We need to collect four groups of data: N_{fb}, N_{fg}, N_{rg} and N_{rb}. To identify the four flows of entities, we add two "Decide" modules named "defergood" and "deferbad". After the two-stage screening processes, entities that the approver think as good applicants, go through "defergood." Its type is 2-way by "Condition," where condition is "$type \leq 0$" In other words, if its "type = 0," then it is a good applicant, called *realgood* here we count it to N_{rg}, on the other hand, if its "type = 1," then it is a bad applicant, called *fakegood*, here we count it to N_{fg}. Entities that approver thinks are bad applicants go through "deferbad." Its type is "2-way by Condition," where condition is "$type \geq 1$" In other words, if "type = 0," then it's a good applicant, called *fakebad*, we count it to N_{fb}, where as if its "type = 1," then it's a bad applicant, called *realgood*, we count it to N_{rb}. Figure 15 takes "defergood" as an example, which shows how to divide the entities.

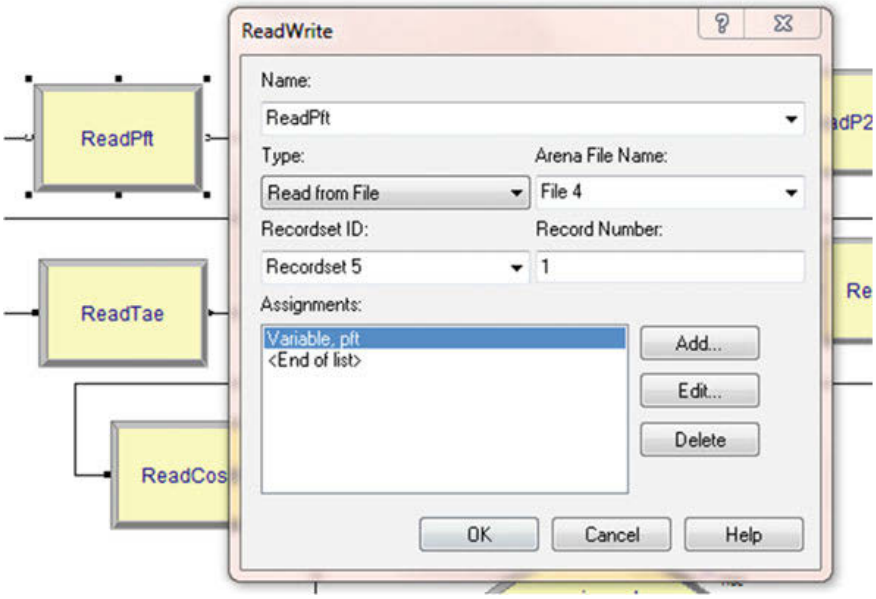


Fig. 12 An example for "ReadWrite" module setting

File - Advanced Process

	Name	Access Type	Operating System File Name	End of File Action	Initialize Option	Recordsets
1	File 1	Microsoft Excel 2007 (*.xlsx)	D:\thesis\documenting\server_parameters.xlsx	Dispose	Hold	3 rows
2	File 2	Microsoft Excel 2007 (*.xlsx)	D:\thesis\documenting\screening_probabilities.xlsx	Dispose	Hold	3 rows
3	File 3	Microsoft Excel 2007 (*.xlsx)	D:\thesis\documenting\applicants_utility_parameters.xlsx	Dispose	Hold	4 rows
4	File 4	Microsoft Excel 2007 (*.xlsx)	D:\thesis\documenting\probability_of_good.xlsx	Dispose	Hold	1 rows

Double-click here to add a new row.

Fig. 13 List of "Files" in Arena

ReadWrite - Advanced Process

	Name	Type	Arena File Name	Recordset ID	Record Number	Assignments
1	ReadPft	Read from File	File 4	Recordset 5	1	1 rows
2	ReadP1	Read from File	File 2	Recordset 9	1	1 rows
3	ReadMu	Read from File	File 1	Recordset 6	1	1 rows
4	ReadRewardg	Read from File	File 3	Recordset 1	1	1 rows
5	ReadP2	Read from File	File 2	Recordset	1	1 rows
6	ReadP3	Read from File	File 2	Recordset	1	1 rows
7	ReadTae	Read from File	File 1	Recordset 7	1	1 rows
8	ReadTbe	Read from File	File 1	Recordset 8	1	1 rows
9	ReadCostw	Read from File	File 3	Recordset 3	1	1 rows
10	ReadRewardb	Read from File	File 3	Recordset 2	1	1 rows
11	ReadCostb	Read from File	File 3	Recordset 4	1	1 rows

Fig. 14 List of "ReadWrite" module in Arena

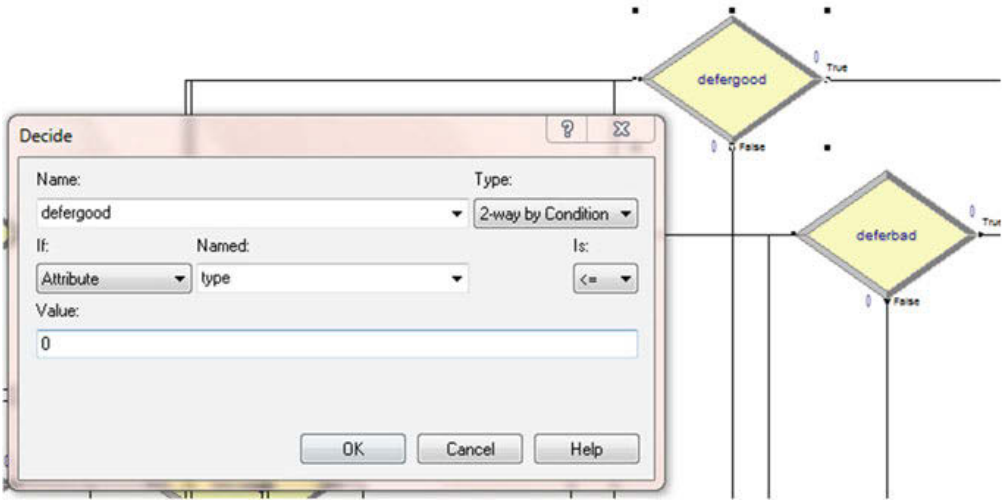


Fig. 15 Dividing entities for four groups of entities

Statistic - Advanced Process

	Name	Type	Expression	Report Label	Output File
1	realgood	Output	defergood.N	realgood	D:\thesis\documenting\realgood.csv
2	fakegood	Output	defergood.N	fakegood	D:\thesis\documenting\fakegood.csv
3	realbad	Output	deferbad.Nu	realbad	D:\thesis\documenting\realbad.csv
4	fakebad	Output	deferbad.Nu	fakebad	D:\thesis\documenting\fakebad.csv

Fig. 16 List of outputs in "Statistic"

To collect the entity number that goes through "defergood" and "deferbad," we use "Statistic" module. In the "Statistic" table, we have four outputs: "realgood, fakegood, realbad, and fakebad." Their type is "Output", and expressions are "defergood.NumberOut True, defergood.NumberOut False, deferbad.NumberOut True, deferbad.NumberOut False." "Output File" will be in a .csv file with the path, as shown in Fig. 16.

3.4 Setting up Simulation

We collect data including N_{fb}, N_{fg}, N_{rg} and N_{rb} based on the difference in screening probabilities to run the simulation. There are $p1$, $p2$ and $p3$ screening probabilities throughout the imperfect two-stage screening system. Starting from 0 %, we take 5 %

each step to reach next level of screening probability. There are total $21 \times 21 \times 21 = 9,261$ sets of screening probabilities that needs to be run. We make use of replications in Arena to use the data from recording. In each replication, we change a set of screening probabilities. The replication length is the simulating experiment period. To make change to set of screening probabilities in every replication, we assign probabilities from particular row in the data column, which is row "c." We define "c" as equal to the current replication number, written as $c = NREP$.

Matlab is used to generate an excel of 9,261 rows × 3 columns of probabilities. We can make another set of $p1$, $p2$ and $p3$ to do the simulation by generating a new set of inputs. This will be explained in Sect. 4. However, if the set of inputs changes, the run times should change as well. Figure 3 shows the simulation of a two-stage imperfect security screening system.

4 Designing a Graphic User Interface with Matlab

After coding, GUI is friendly for users to complete the tasks. We add the following functions to GUI: generating input for simulation, pulling data from simulation results, analyze data, and generate graphs.

First, we design the function modules to generate potential applicants arrival rate and service rate. Second, we design module to pull data from simulation and to find optimization and its condition. Third, we design an analyzing module for sensitive analyses. Last but not least, we design a record module for analyzed data. To realize these functions, we first create a new GUI, which goes to "HOME → New → Graphical User Interface" and naming it *screening*. Then, we divide the Blank GUI figure into 3 function areas by adding 3 "Button Group," named "Generating Inputs," "Optimization," and "Data Analysis."

4.1 Generating Input Parameters for Simulation

The input we need to generate includes the following: screening service rate μ, reward and cost for good applicants and bad applicants r_g, r_b, c_b, c_w, *type I and type II* error α, β, percentage of good applicants in total P, and screening probabilities $\Phi_1, \Phi_{2g}, \Phi_{2b}$. We use "static txt" to label inputs and "edit txt" for user to specify inputs. In total, 23 "static txt"s and 11 "edit txt"s are added to "Generating Inputs" Button Group in Fig. 17a.

Fig. 17 Generating inputs with approver's preference in GUI. **a** Generating inputs in GUI. **b** Defining the Approver's Preference

4.2 Calculating the Optimal Strategies Using Numerical Methods

In this section, we pull raw data and check their usability and then profile them. We add 4 "edit box"s to define approver's preference R_g, R_b, C_g and C_b for "Optimization" as shown in Fig. 17b.

Data reflecting N_{fb}, N_{fg}, N_{rg} and N_{rb} are saved in a .csv file. We find the raw data from Arena that have even rows writing 0 and odd rows writing data, which we need to profile.

After profiling the data, we can generate a matrix of utility according to different set of data. Then, we use "Max" function to find the optimization set of data from the matrix. Results of optimization will be shown on screen. In the meantime, a graph reflecting all the data in matrix is drawn to show how screening probability affects the approver's utility. We use green dots to show all the data points, and red diamond to point out the best strategy.

4.3 Designing Output Data Analysis

Sensitivity analysis is the study of how the uncertainty in the output of a mathematical model or system (numerical or otherwise) can be apportioned to different sources of uncertainty in the input. We want to see, once Φ_1, Φ_{2g} or Φ_{2b} is fixed to 100 %, how the other screening probabilities affect the approver's utility and the

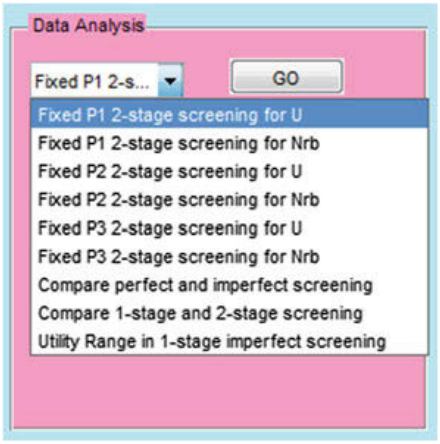


Fig. 18 Data visualization and analysis in Arena

number of bad applicants getting caught. We add an Axe in GUI and put "Pop-up Menu" in "Data Analysis" function area. "Pop-up Menu" can give cases which we can call by adding a push button "GO" in its callback. Figure 18 shows the "Data Analysis" Part.

The layout of GUI is shown below in Fig. 19.

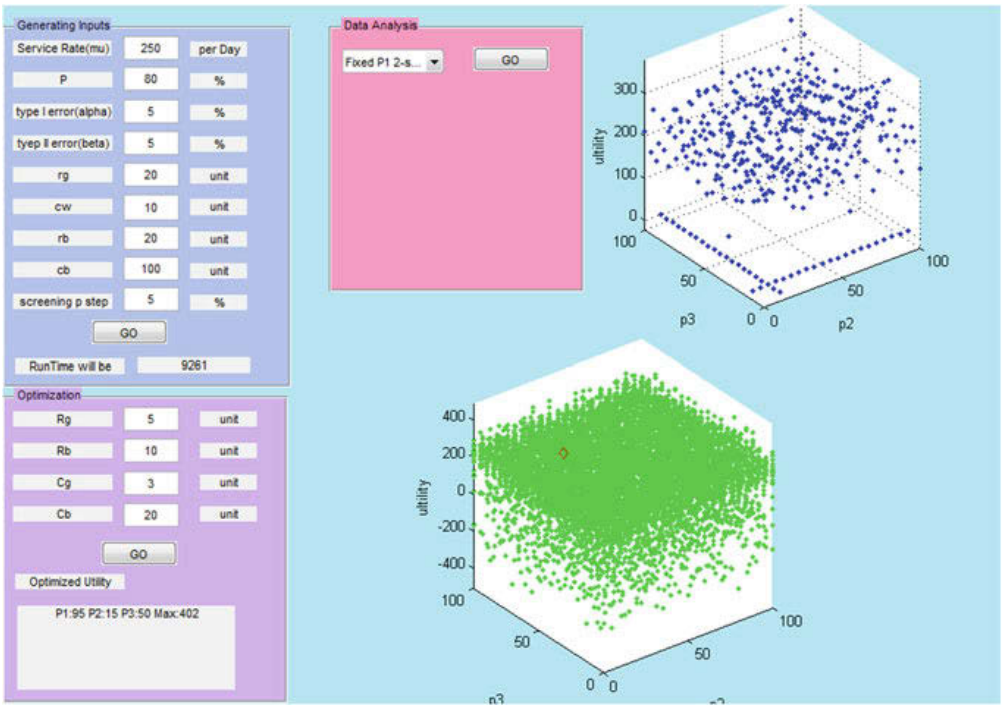


Fig. 19 Layout of the matlab GUI

5 Numerical Experiments

5.1 Data Sources for Input Parameters

We use the baseline according to the paper [27] to do a new numerical experiment. For good applicants, we have $r_g = 20$ and $c_w = 10$. For bad applicants, we have $r_b = 20$ and $c_b = 100$. For approver, we have $R_g = 5$, $R_b = 10$, $C_g = 3$, and $C_b = 20$. Using the data from Newark Liberty International Airport [18], we estimate the arrival rate and the service rate. The average number of arriving passengers is in Eq. (13).

$$\begin{aligned} N_{\text{arrive per year}} &= (34014027 + 33711372 + 33107041 + 33424110 + 35366359 \\ &\quad + 36367240 + 35764910 + 33078473 + 31893372 + 29428899 \\ &\quad + 29220775 + 31100491 + 34188701 + 33622686 + 32575874 \\ &\quad + 30945857 + 26626231 + 22255002)/(18) \\ &= 32038400 \end{aligned} \tag{13}$$

There are three terminals Terminal A, B, and C in this airport. Usually, five securities opening in a day in each terminal are expected. To simulate $M/M/1$ queue, the arrival rate for modeling can be defined in Eq. (14).

$$\Lambda = \frac{N_{\text{arrive per year}}}{\text{days of a year} \times \text{servers}} = \frac{32038400}{365 \times 15} = 5851.76 \approx 5852 \tag{14}$$

In Sect. 3.1, we have $1/5852 \approx 0.00017$, then the arrival setting is POIS (0.00017), Unit is "Days". We round it down to 0.0001 and round it up to 0.0002, to run twice.

According to experiences from airport security screening, the average screening time is 5 min per person. We estimate the service rate following the equation below:

$$\mu = \frac{\text{Minutes in a day}}{\text{Service Time per Person}} = \frac{60 \times 24}{5} = 288 \tag{15}$$

Because $1/288 \approx 0.0034$, in Sect. 3.1, the service setting in screening is EXPO (0.0034), Unit will be "Days". We round it up to 0.004 and round it down to 0.003, to run twice.

We have two sets of Λ and μ to do the numerical experiment. They are $\Lambda = 10,000$, $\mu = 250$ with $\frac{\Lambda}{\mu} = 40$ and $\Lambda = 5000$, $\mu = 333$ with $\frac{\Lambda}{\mu} = 15$. We record entities going through the module to record data in a .csv file. We do 9,261 replications to set up different combinations of probabilities for each simulation and the replication length is 30 days.

5.2 Simulation Results: Optimal Strategies and Payoffs

We set $\frac{\lambda}{\mu} = 40$, where $\Lambda = 10,000$ and $\mu = 250$. We put arrival as POIS(0.001) and service as EXPO(0.04), where we have the best screening strategies for two-stage imperfect screening process with $\Phi_1 = 10\,\%, \Phi_{2b} = 10\,\%, \Phi_{2g} = 15\,\%$ and $U = 9,505$.

Red Diamond point shows the best strategy point. Figure 20 shows how Φ_{2b} and Φ_{2b} affect the approver's utility. Figure 20a shows an interesting jump when $\Phi_{2g} = \Phi_{2b}$. It seems that when the second stage has the equal probability of screening for potential good applicants and potential bad applicants, it's the worst case with smallest approver's utility. It neither attract potential applicants nor does any good to the approver. When $\Phi_{2g} > \Phi_{2b}$, the utility is greater than the case of $\Phi_{2b} > \Phi_{2g}$. In this case, approver should put much effort on screening probability in Φ_{2g}.

We set $\frac{\lambda}{\mu} = 15$, where $\Lambda = 5,000, \mu = 333$. We put arrival as POIS(0.002) and service as EXPO(0.03), where we have the best strategies for two-stage imperfect screening process with $\Phi_1 = 100\ \%$, $\Phi_{2b} = 0\ \%$, $\Phi_{2g} = 0\ \%$, and $U = 10,430$. Figure 20b shows an interesting jump when $\Phi_{2g} = \Phi_{2b}$. It seems that when the second stage has the equal probability of screening for potential good applicants and potential bad applicants, it is the worst case with smallest approver's utility. It does not attract potential applicants or does any good to the approver. When $\Phi_{2g} > \Phi_{2b}$, the utility is greater than the one when $\Phi_{2g} > \Phi_{2b}$. Approver should put much effort on screening probability Φ_{2g}. We find that with two different ratio of λ and μ, the results are consistent. We can conclude that when $\Phi_{2g} = \Phi_{2b}$ is the worst case for approver; Since second-stage screening, Φ_{2g} is more important, on which approver should focus more.

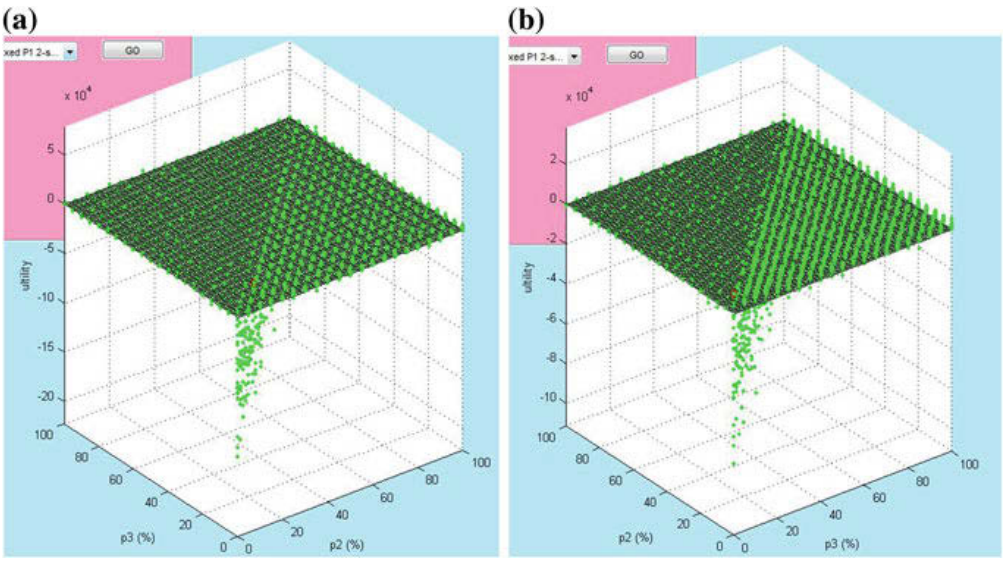


Fig. 20 Illustration of approver's utility affected by screening probabilities $\Phi_{2g}(P_3)$ and $\Phi_{2b}(P2)$. **a** $\lambda/\mu = 40$, **b** $\lambda/\mu = 15$

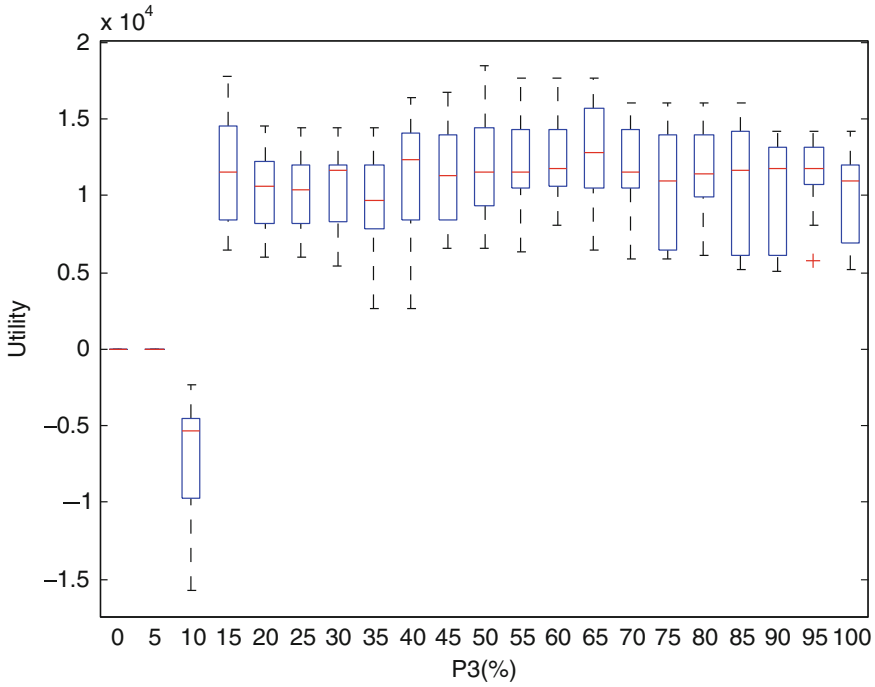


Fig. 21 Illustration of approver's utility affected by screening probabilities $\Phi_{2g}(P3)$, fixing $\Phi_{2b} = 10\,\%(P2)$ and $\Phi_1 = 10\,\%(P1)$ with $\frac{\lambda}{\mu} = 40$

In addition, we do another simulation to see how Φ_{2g} impacts utility when fixing $\Phi_1 = 10\,\%$, and $\Phi_{2b} = 10\,\%$. This is based on the optimal strategy for approver when $\frac{\lambda}{\mu} = 40$. Figure 21 shows that when $\Phi_{2g} \leq 15\,\%$, approver's utility ≤ 0; when $\Phi_{2g} > 15\,\%$, approver's utility > 0. In this case, if the approver does not want to screen all applicants, they can screen 15 % of good applicants at the second stage, who are defined at the first stage. The results are based on this particular set of parameters. Changing input parameters may change the results accordingly.

6 Conclusion and Future Research Directions

In this research, we develop several modules in both Arena and Matlab to simulate and analyze an imperfect multi-stage screening system with screening errors. By setting conditions for decision modules and locating modules in different positions, we are able to control the applicants' flow in the system according to prespecified conditions. We use different settings and positions of decision nodes to control the arrival rate, classify good and bad applicants, and simulate type I and type II errors. We use replications to repeat simulation and get statistically significant results, with different sets of input to the same model. We can access and control the screening process based on the user/approver's preferences. By considering the potential

applicants best responses and rates of arrival and service, the approver is able to find the best screening strategy to maximize her utility, based on simulation results.

To our best knowledge, this is the first research using simulation, queueing theory, and game theory to study a complex multi-stage screening system. In the future, we could extend this work to study more complex models with various distributions of approval and service processes. Besides, we can extend the simulated model as multi-stage screening imperfect model. On the other hand, the fast development of smart phones and social media makes it possible to use dynamic and real-time data to simulate and update the security screening process. For example, there is a recent smart phone-based app which enables passengers to post their waiting time in line for security screening at airports [26]. This enables both other passengers and the approver to get more accurate and timely information about the security screening. Future research could model and simulate dynamic systems, considering waiting time for the bad applicants.

References

1. Bennett AC, Chin YZ (2008) 100 % container scanning: security policy implications for global supply chains. Master's thesis, Massachusetts Institute of Technology, Engineering Systems Division
2. Bier VM, Haphuriwat N (2011) Analytical method to identify the number of containers to inspect at us ports to deter terrorist attacks. Ann Oper Res 187(1):137–158
3. Directorate-General Energy and Transport (2009) The impact of 100 transport. http://ec.europa.eu/transport/modes/maritime/studies/doc/2009_04_scanning_containers.pdf. Accessed July 2014
4. Dorton SL (2011) Analysis of airport security screening checkpoints using queuing networks and discrete event simulation: a theoretical and empirical approach. Embry-Riddle Aeronautical University, Daytona Beach
5. Eckel N, Johnson W (1983) A model for screening and classifying potential accounting majors. J Acc Educ 1(2):57–65
6. Fleming A (2014) Oversold flights, getting bumped and bumping. http://airtravel.about.com/od/travelindustrynews/qt/bumped1.htm. Accessed July 2014
7. Gasson S, Shelfer KM (2007) IT-based knowledge management to support organizational learning: visa application screening at the INS. Inf Technol People 20(4):376–399
8. Gaukler GM, Li C, Ding Y, Chirayath SS (2011) Detecting nuclear materials smuggling: performance evaluation of container inspection policies. Risk Anal 32(3):65–87
9. Gilliam RR (1979) Application of queuing theory to airport passenger security screening. Interfaces 9(4):117–123
10. Kelton WD, Sadowski RP, Swets NB (2010) Simulation with arena. McGraw-Hill, Higher Education, Boston
11. Kronik L, Shapira Y (1999) Surface photovoltage phenomena: theory, experiment, and applications. Surf Sci Rep 37(1):1–206
12. Maddox MW, Gubbins KE (1997) A molecular simulation study of freezing/melting phenomena for lennard-jones methane in cylindrical nanoscale pores. J Chem Phys 107 (22):9659–9667
13. MathWorks (2014) MATLAB GUI. http://www.mathworks.com/discovery/matlab-gui.html. Accessed July 2014

14. McCarley JS, Kramer AF, Wickens CD, Vidoni ED, Boot WR (2004) Visual skills in airport-security screening. Psychol Sci 15(5):302–306
15. Merrick JRW, McLay LA (2010) Is screening cargo containers for smuggled nuclear threats worthwhile? Decis Anal 7(2):155–171
16. Musa JD (2004) Software reliability engineering: more reliable software, faster and cheaper. Tata McGraw-Hill Education, New York
17. Myerson RB (1997) Game theory: analysis of conflict. Harvard University Press, Cambridge
18. Newark Liberty International Airport (2014) Facts and information. http://www.panynj.gov/airports/ewr-facts-info.html. Accessed July 2014
19. Pendergraft DR, Robertson CV, Shrader S (2004) Simulation of an airport passenger security system. In: Proceedings of the 36th conference on Winter simulation, pp 874–878
20. Regattieri A, Gamberini R, Lolli F, Manzini R (2010) Designing production and service systems using queuing theory: principles and application to an airport passenger security screening system. Int J Serv Oper Manage 6(2):206–225
21. Roxy P, Devore JL (2011) Statistics: the exploration and analysis of data. Oxford University Press, Oxford
22. Shustorovich E (1986) Chemisorption phenomena: analytic modeling based on perturbation theory and bond-order conservation. Surf Sci Rep 6(1):1–63
23. Sundarapandian V (2009) Queueing theory. probability, statistics and queueing theory. PHI Learning, New Delhi
24. Tadelis S (2013) Game theory: an introduction. Princeton University Press, Princeton
25. Transportation Security Administration (2013a) Certified cargo screening program. http://www.tsa.gov/certified-cargo-screening-program. Accessed July 2014
26. Transportation Security Administration (2013b) My TSA mobile application. http://www.tsa.gov/traveler-information/my-tsa-mobile-application. Accessed July 2014
27. Wang X, Zhuang J (2011) Balancing congestion and security in the presence of strategic applicants with private information. Eur J Oper Res 212(1):100–111
28. Wolfram S (2002) Some historical notes. http://www.wolframscience.com/reference/notes/971c Accessed July 2014
29. Yoon D (2007) Analysis of truck delays at container terminal security inspection stations. Ph.D. thesis, New Jersey Institute of Technology, New York
30. Zeltyn Marmor YN, Mandelbaum et al SA (2011) Simulation-based models of emergency departments: operational, tactical, and strategic staffing. ACM Trans Model Comput Simul (TOMACS) 21(4):24
31. Zhuang J, Bier VM (2007) Balancing terrorism and natural disasters–defensive strategy with endogenous attacker effort. Oper Res 55(5):976–991