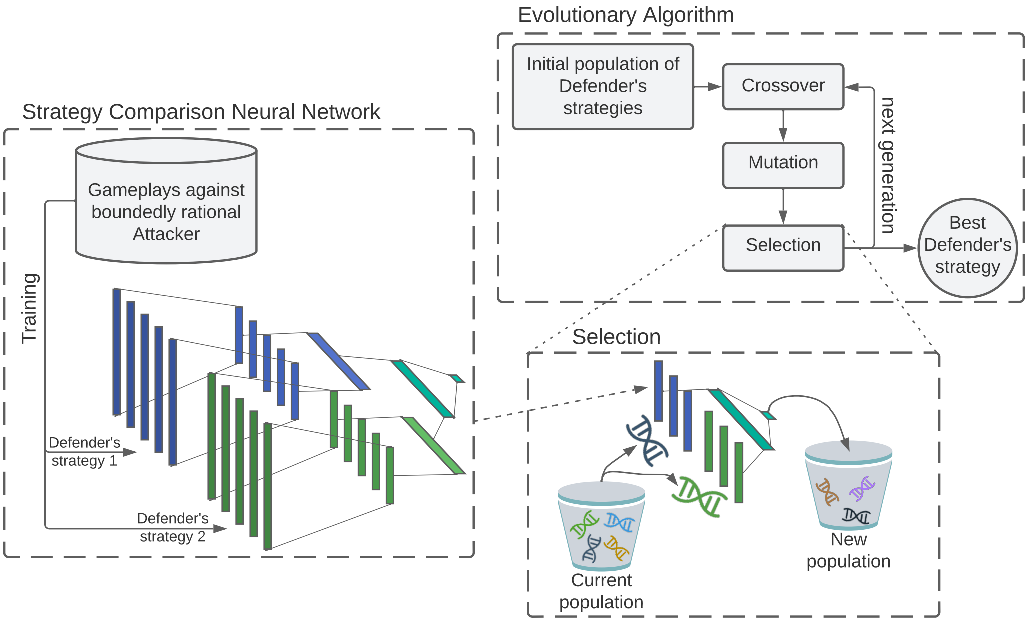
Graphical Abstract

Duel-based neuroevolutionary method for Stackelberg Security Games with boundedly rational Attacker

Jacek Mańdziuk, Adam Żychowski



Duel-based neuroevolutionary method for Stackelberg Security Games with boundedly rational Attacker

Jacek Mańdziuk[a,b,*], Adam Żychowski[a]

[a] *Faculty of Mathematics and Information Science, Warsaw University of Technology, 00-662 Warsaw, Poland*

[b] *Institute of Computer Science, AGH University of Science and Technology, 30-059 Krakow, Poland*

Abstract

This paper considers the problem of finding optimal strategies in Stackelberg Security Games when playing against a non-perfectly rational Attacker. To this end, a novel *Duel-based NeuroEvolutionary approach to Security Games* (DNESG) is proposed, which utilizes the *Strategy Comparison Neural Network* (SCNN) as a surrogate model to compare pairs of Defender's strategies. SCNN is trained on historical data (past attack attempts) and does not require any direct information about the Attacker's preferences regarding targets, payoff distribution, or decision-making model. SCNN is embedded in the Evolutionary Algorithm framework and implements a tournament-based selection method in place of a time-consuming direct strategy evaluation. The effectiveness of DNESG is assessed on a set of 90 benchmark *Deep Packet Inspection* games inspired by real cybersecurity scenarios. The proposed method provides high-quality solutions and outperforms state-of-the-art approaches (both exact and approximate) with statistical significance

[*] Corresponding author

Email address: mandziuk@mini.pw.edu.pl (Jacek Mańdziuk)

when playing against non-perfectly rational Attacker. Moreover, DNESG offers excellent time scalability, being two orders of magnitude faster than the state-of-the-art Mixed-Integer Linear Programming method.

Keywords: Neuroevolution, Stackelberg Equilibrium, Security Games, Bounded Rationality, Evolutionary Computation, Game Theory, Cybersecurity

1. Introduction

Decades of game theory research have helped to develop more informed decision-making processes. This is particularly crucial in the context of public safety, including protection against theft, terrorism, and cyberattacks. One specific branch of game theory, known as Stackelberg Security Games (SSGs) [1], has gained significant popularity for its ability to model such types of public safety scenarios. Due to their practical relevance, SSGs have been vastly influential in security research in recent years.

SSGs are a class of game models with a wide range of practical applications in many domains such as homeland security [2], natural environment protection [3], power markets bidding [4], e-commerce supply chain sustainability [5], or cybersecurity [6]. SSGs are played by two players: the Defender and the Attacker. The Defender commits to his/her strategy first, and subsequently, the Attacker chooses his/her strategy based on his/her preferences and/or payoffs, and taking into account the Defender's strategy. While choosing the strategy, the Attacker is aware of the Defender's *mixed strategy* only, represented as a probability distribution of the Defender's *pure strategies*. The Attacker does not know the specific Defender's strategy materialization.

The objective of the game is to find the Defender's strategy that maximizes his/her expected payoff. The problem of finding the optimal strategy for the Defender has been proven to be NP-hard [7].

One of the underlying principles of the vast majority of SSG formulations considered in the literature is the assumption of perfect rationality of the players, which means that both players would always choose strategies that are optimal for them. However, many SSGs have been formulated in response to practical needs for modeling security-related situations. Such systems are usually designed to help in choosing the optimal strategy of the Defender, who can be, for instance, airport guards, security companies, or police. Their opponents are terrorists, thieves, criminals, etc., whose decisions may be suboptimal for various reasons (e.g. stress, insufficient knowledge, or time pressure). Therefore, it is important that this type of *decision imperfection* of the Attacker are taken into account when designing security systems. In game theory, this imperfection is referred to as *bounded rationality* (BR) [8].

Previous research in the area of Security Games demonstrated that incorporating BR models into the process of computing an optimal Defender's strategy can lead to improved performance when playing against a human Attacker [9, 10, 11, 12, 13]. However, the selection of a specific BR model is not always clear, and in many cases, little information is available about the personality or preferences of the Attacker [14]. To address this issue, instead of assuming a particular BR model *a priori*, we aim to *learn* the Attacker's decision-making model from the past data.

Specifically, we design a neural network able to compare two Defender's strategies. The network is embedded in the evolutionary framework [15] and used multiple times in the chromosome evaluation phase. The proposed solution does not require any prior knowledge about the Attacker, as it learns solely from historical data (i.e. previous attack attempts).

The algorithm uses a population of individuals, each of which encodes a candidate Defender's strategy. The population is evolved over a fixed number of iterations, during which current strategies are modified by evolutionary operators – mutation and crossover. The resulting strategies are then indirectly evaluated in the selection procedure in the form of a binary tournament, based on the neural network output. The promoted individuals constitute the next generation.

1.1. Contribution

The main contribution of this paper is threefold:

- A novel duel-based neuroevolutionary algorithm for solving Stackelberg Security Games with non-perfectly rational Attacker (DNESG) is proposed.

- The key component of DNESG is a neural network model used in a binary tournament evaluation, able to accurately compare Defender's strategies and leading to overall high quality solutions.

- In effect, an end-to-end method for learning the Attacker's decision-making bounded rationality model based on past data (previous attacks) is obtained. The method uses this information to evolve highly efficient Defender's strategies.

This study extends our previous conference paper [16], in which a similar neuroevolutionary system (NESG) was proposed, differing in the following four aspects:

- Instead of a direct assessment of the chromosomes (candidate Defender's strategies) introduced in [16], a binary tournament evaluation is proposed, using specifically trained neural network surrogate model (SCNN) that compares a pair of strategies provided in the input.
- A selection phase and an elite mechanism are modified accordingly to utilize the outcomes of a series of SCNN duels when creating a new generation.
- The number of past attack episodes (learning samples) required for efficient approximation of the Attacker's decision-making model is significantly smaller compared to [16], due to different evaluation scheme (a series of binary tournaments proposed in this paper versus a direct chromosome assessment implemented in [16]).
- The role of the neural network component in the whole neuroevolutionary method is investigated in detail.

The remainder of this paper is arranged as follows. Section 2 presents definitions of bounded rationality models used in the experiments, a formulation of Stackelberg Equilibrium, a description of the considered application problem from the domain of cybersecurity, and a motivating example game. An overview of related work devoted to solving SSGs with bounded rationality is presented in Section 3. Section 4 provides a detailed description of the proposed surrogate-assisted neuroevolutionary algorithm (DNESG), tailored for computationally expensive SSGs. In Section 5, the experimental evaluation of DNESG is carried out on a suite of test games with varying degree of complexity. In the next section, the results are discussed in the context of SCNN accuracy, obtained payoffs, as well as robustness and time scalability of DNESG. The last section is devoted to conclusions and directions for future research.

2. Definitions

2.1. Bounded rationality

The term *bounded rationality* (BR) was coined in 1957 by Herbert Simon in the book *Models of Man* [8]. Initially, this topic did not receive much attention but gained momentum in 1990s [17] when the use of game theory models in practical applications became popular and the necessity to find models that best reflect reality, i.e. take into account the imperfections of player's decisions, arose.

It should be noted that bounded rationality is not equivalent to irrationality. The concept of irrationality or lack of rationality refers to situations in which player's actions are unpredictable or illogical, whereas bounded rationality assumes that the player is trying to choose the optimal strategy but for some reason is unable to do so. The main hindering factors include limited cognitive abilities, lack of skills to accurately evaluate a given situation, limited time to make a decision, or unpredictable circumstances. Many works in the field of psychology indicate that people have a tendency to simplify reality and choose easier paths than the optimal ones. In [18] they show that mental saving does not stem from laziness but prevents the system from being overloaded. As the complexity of the surrounding world increases, the need for *shortcuts* increases, and the loss caused by simplifications and non-optimal decisions is lower than the cost of the effort put into deeper analysis.

Over the years, a handful of popular models of BR were proposed that refer to different aspects of the human cognitive imperfection and reflect the general ways how people make decisions. The choice of a particular BR model often depends on the problem considered, the circumstances in which it is solved, or personal traits of a decision maker. Among the most popular, there are three BR models considered in this paper: Anchoring Theory, Quantal Response, and Prospect Theory.

2.1.1. Anchoring Theory

Anchoring Theory (AT) [19] postulates that people have a tendency to flatten the probabilities of presented choices. In the decision-making process, the probability distribution is perceived as being closer to a uniform distribution than it is in reality. This means that high probabilities of events are perceived as lower and small probability values are perceived as higher. Formally, this dependence can be described as follows:

$$p'(x) = p(x)(1 - \delta) + \frac{\delta}{|X|}, \tag{1}$$

where X is the set of all events, $|X|$ denotes the cardinality of this set, $p(x)$ is the actual probability of event $x \in X$, $p'(x)$ is the disturbed probability (perceived by people according to the AT), and $\delta \in [0, 1]$ is a parameter determining the strength of the disturbance. For $\delta = 0$ we have $p' = p$, which means that perceived probabilities are equal to the actual ones, while for $\delta = 1$: $\forall_{x \in X} p'(x) = \frac{1}{|X|}$, which means a uniform probability distribution. In this paper a value of $\delta = 0.5$ is assumed, which is most commonly encountered in the literature.

2.1.2. Prospect Theory

Prospect Theory (PT) [20] relies on the observation that the aversion to losses and the desire for gains are asymmetrical. In psychological experiments, it was shown that people have a strong aversion to taking risks that could result in significant losses, even in the face of the possibility of making great gains. In other words, they preferred to participate in less risky bets even if the expected values were lower.

The experiments also showed that instead of maximizing the expected payoff, people subconsciously maximize the so-called *prospect* (P), which can be defined as follows:

$$P = \sum_i f(p_i) g(u_i), \tag{2}$$

$$f(p_i) = \frac{p_i^\gamma}{(p_i^\gamma + (1 - p_i)^\gamma)^{\frac{1}{\gamma}}}, \quad g(u_i) = \begin{cases} u_i^\alpha & \text{for } u_i \geqslant 0 \\ -\theta \cdot (-u_i)^\beta, & \text{for } u_i < 0 \end{cases} \tag{3}$$

where f and g are functions that transform the perception of the actual probability p_i of obtaining payoff u_i. γ, θ, α and β are parameters. In this study, in accordance with the recommendations presented in [21], the following parameter values are adopted: $\gamma = 0.64, \theta = 2.25, \alpha = \beta = 0.88$. Figure 1 visualizes both functions.

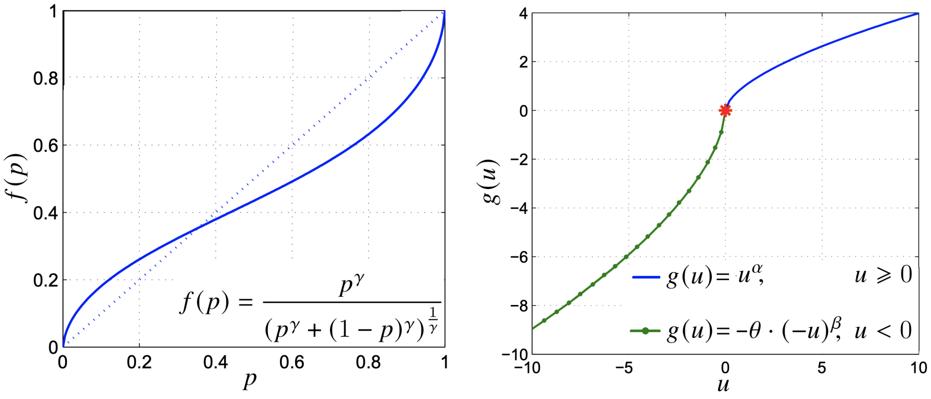


Figure 1: Functions representing the perception of probability (on the left) and payoff (on the right) according to the Prospect Theory.

2.1.3. Quantal Response

The theory of *Quantal Response* (QR) [22] states that people make decisions randomly but with some dependence on the payoff, meaning that the greater the payoff associated with a certain decision, the higher the probability of choosing it. This means that the optimal decision has the highest probability of being chosen, but any other decision can also be made. According to this theory, the probability of making a decision x_i is expressed by the following formula:

$$p(x_i) = \frac{e^{\lambda u(x_i)}}{\sum_{x_j \in \Sigma} e^{\lambda u(x_j)}}, \tag{4}$$

where $u(x_i)$ is the expected payoff of decision x_i, Σ is the set of all possible decisions, and $\lambda \geqslant 0$ is a parameter. If $\lambda = 0$, the choice is completely random - every decision is equally probable. For $\lambda \to \infty$ the optimal decision is definitely made - a fully rational choice. In the experiments described in this paper, following [23], $\lambda = 0.8$ was adopted.

2.2. Stackelberg Equilibrium

In Stackelberg Security Games there are two non-cooperative players: the Defender (D) and the Attacker (A). Their knowledge of the opponent's strategy is asymmetric. The Defender chooses his/her strategy first, and then the Attacker, knowing the Defender's choice, decides about his/her strategy. This implies certain advantage of the Attacker in terms of possessed information. However, the Defender does not have to choose a pure strategy – usually he/she chooses a mixed strategy. In effect, the Attacker is aware of the probabilities with which the Defender will choose each of his/her pure strategies, but does not know the exact Defender's decision regarding the selected strategy.

Stackelberg Equilibrium (SE) is defined as a pair of player's strategies $(\bar{\pi}^D, BR(\bar{\pi}^D))$ that satisfy the following conditions:

$$\bar{\pi}^D = \underset{\pi^D \in \Pi^D}{\arg\max} \quad U^D(\pi^D, BR(\pi^D)), \tag{5}$$

$$BR(\pi^D) = \underset{\pi^A \in \Pi^A}{\arg\max} \quad U^A(\pi^D, \pi^A) \tag{6}$$

where π^D and π^A denote the Defender and the Attacker strategy, respectively, and Π^D, Π^A are sets of their all possible strategies. $BR(\pi^D)$ is the optimal Attacker's response (maximizing his/her payoff) to the strategy π^D of the Defender. $U^p(\pi^D, \pi^A)$ denotes the payoff of player $p \in \{D, A\}$ when strategies π^D and π^A are selected, respectively by the Defender and the Attacker.

Equation 5 defines the optimal strategy $\bar{\pi}^D$ for the Defender, under the assumption that the Attacker will always choose the best response strategy.

Please observe that selection of $BR(\pi^D)$ by the Attacker in Eq. 6 may not be unique, i.e. there may exist more than one best Attacker's response to a given strategy π^D of the Defender. In accordance with the definition of Strong Stackelberg Equilibrium (SSE) [24], if there exist multiple optimal strategies for the Attacker in Eq. 6 (with the same highest payoff), among them the Attacker will select a strategy that results in the highest payoff for the Defender, i.e. the Attacker will break ties in favor of the Defender. While this assumption may appear counterintuitive, the opposite approach may result in SE non-existence [25]. SSE is widely adopted in the SSG literature and is also considered in this paper.

It is worth noting that both players commit to their strategies at the beginning of the game, before the first moves are played, and are unable to alter them during the gameplay. This implies that throughout the game, they will take actions according to their chosen strategies, regardless of the opponent's moves, as they are not privy to the actions performed by the adversary.

2.3. Problem definition

We consider an m-step SSG with a predefined set of n targets $T = \{t_1, t_2, \ldots, t_n\}$. Each target $t \in T$ has 4 associated payoffs: U_t^j, $j \in \{D+, D-, A+, A-\}$ representing the Defender's reward (U_t^{D+}), their penalty (U_t^{D-}), the Attacker's reward (U_t^{A+}), and his/her penalty (U_t^{A-}). $U_t^{D+} > U_t^{D-}$ and $U_t^{A+} > U_t^{A-}$.

The Defender possesses k units. A Defender's *pure strategy* σ^D defines an allocation of units to targets in each of m time steps. Units can be reallocated between the steps. Formally $\sigma^D = \{a_{us}\}$, where $a_{us} \in T$ is a target allocated to unit u in time step s, $u \in \{1, \ldots, k\}, s \in \{1, \ldots, m\}$.

Let's denote a set of all pure strategies of the Defender by Σ^D. Then, a *mixed strategy* π^D is a probability distribution over Σ^D, i.e. $\pi^D = \{(\sigma_i^D, p_i)\}$, where p_i is the probability of playing strategy $\sigma_i^D \in \Sigma^D$.

A coverage of target t *in step* s ($c^s(t)$) associated with the mixed strategy π^D is defined as a probability of the event that at least one unit is allocated to t in step s when strategy π^D is played, i.e. $c^s(t) = \sum_{\sigma_i^D \in \pi^D} p_i : \underset{a_{us} \in \sigma_i^D}{\exists} a_{us} = t$.

The Attacker's strategy σ^A consists in choosing one of the targets x from T. The players' payoffs are computed as follows:

- If in any time step any Defender's unit is allocated to x, the Attacker is *caught* and the players receive $U_x^{D^+}$ and $U_x^{A^-}$, respectively.

- If none of the Defender's units is allocated to x across all time steps, the attack is successful and the players receive $U_x^{D^-}$ and $U_x^{A^+}$, respectively.

Therefore, the expected players' payoffs (U^D and U^A) are equal to

$$U^D = P_x U_x^{D-} + (1 - P_x) U_x^{D+}, \tag{7}$$

$$U^A = P_x U_x^{A+} + (1 - P_x) U_x^{A-}, \tag{8}$$

where $P_x = \prod_{s=1,\ldots,m}(1 - c^s(x))$ is a probability of successful attack on target x.

The game model employs Stackelberg Game principles, which means that first the Defender commits to his/her strategy π^D (probability of units allocation) and then the Attacker, being aware of π^D, determines his/her strategy (target selection).

2.4. Cybersecurity scenario

In cybersecurity, a popular approach to secure computer networks is through *deep packet inspections* (DPI) [26]. This method involves a periodic selection of a subset of packets for inspection. This problem can be formulated as a Stackelberg Game, in which the detection system acts as the Defender and the intruder plays the role of the Attacker. The network computers (hosts) are the targets. The detection system selects a subset of hosts and inspects packets sent to them for a fixed period of time, then moves on to the next subset of hosts in the next time step. If malicious packets go undetected, the attack is successful and the intruder controls the infected host. DPI can cause unwanted latency, and the Defender must decide where to inspect network traffic in order to maximize the probability of successful detection of malicious packets.

It is possible to detect DPI by monitoring network traffic for patterns that indicate DPI is being used. For example, if certain types of traffic are blocked or slowed down, that could be a sign that DPI is being used. Hence, the Attacker, by probing hosts can approximate the Defender's strategy.

While the Defender has no direct knowledge about a potential intruder, historical data or simulations can be used to approximate his/her preferences or capabilities.

2.5. Example game

To illustrate the considered problem, in this section we present an example of a simple game and calculate the outcomes of 3 BR models described in Section 3.2, in each case assuming that a particular BR model affects the Attacker's decision-making process. Figure 2 depicts a basic variant of the game with no BR assumption. There are $n = 5$ targets and $k = 2$ Defender's units. For simplicity, we consider a one-step game ($m = 1$) and $\forall t \in T\ U_t^{D+} = 0$. The remaining payoffs $(U_t^{D-}, U_t^{A-}, U_t^{A+})$ are presented in the figure. Next to each target, its coverage ($c^1(t)$) is presented. With Eq. 8 we can compute the Attacker's expected payoff when he/she decides to attack a given target. The highest value is $U_5^A = 0.29$ which determines that the Attacker will choose target T5 which, according to Eq. 7, leads to the expected Defender's payoff $U^D = (1 - 0.2) \cdot (-0.3) = -0.24$.

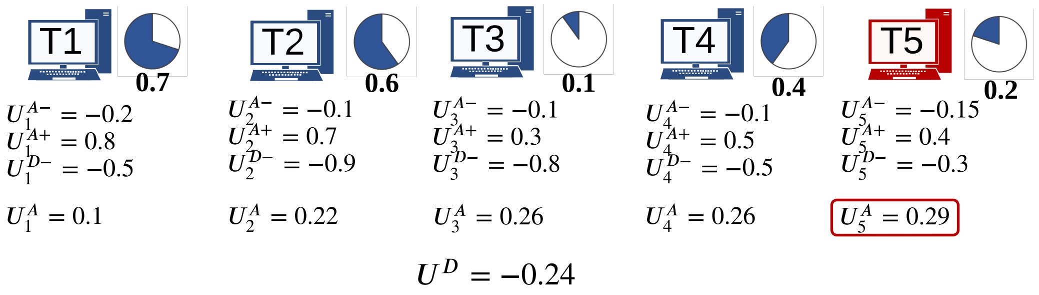


Figure 2: Example game with 5 targets. **No bounded rationality model** is assumed. Right to each target, its coverage is presented. The highest Attacker's payoff is for target T5 and equals 0.29. Therefore, the Defender's expected payoff is equal to -0.24.

The above result assumes that the Attacker is perfectly rational and makes an optimal choice. Now, let us look how different BR models will affect the Attacker's perception and consequently the Defender's payoff.

Figure 3 presents the same game but with the assumption that the Attacker adopts the Anchoring Theory model (see Eq. 1). In this case, the Attacker perceives targets coverage differently (closer to a uniform distribution) and now target T2 is the best choice from his/her point of view. This results in the Defender's expected payoff $U^D = (1 - 0.6) \cdot (-0.9) = -0.36$

(note that when calculating the Defender's payoff, the real, not disturbed coverage is considered). Now, U^D has significantly lower value than in the previous case (without BR model). If the Defender knew how the Attacker perceives the targets' coverage, he/she could alter this coverage to increase his/her expected payoff.

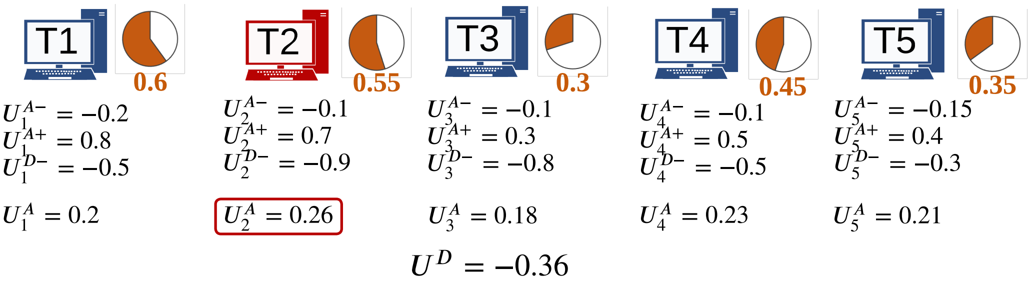


Figure 3: Example game with the assumption that the Attacker adopts the **Anchoring Theory** decision-making model. Next to each target, a disturbed (according to Eq. 1) coverage is presented. From the Attacker's perspective (following AT model) the best choice is target T2. Therefore, the Defender's expected payoff equals -0.36.

If the Attacker follows Prospect Theory his/her perception of both the target coverage and the payoff distribution are disturbed according to Eq.3. Therefore, the most attractive target from the Attacker's perspective is now T3. This implies that the Defender's expected payoff U^D is equal to $U^D = (1 - 0.1) \cdot (-0.8) = -0.72$. The respective game with the disturbed values is depicted in Figure 4).

The last considered BR model is Quantal Response. It assumes that the Attacker, instead of choosing a single target, will attack each target with a certain probability defined in Eq. 4. The respective game is presented in Figure 5. Hence, the expected Defender's payoff U^D equals $0.18 \cdot (1 - 0.7) \cdot$

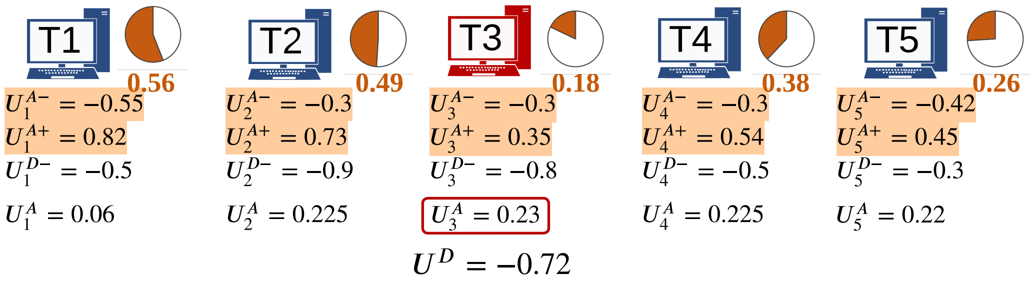


Figure 4: Example game with the assumption that the Attacker adopts the **Prospect Theory** decision-making model. Next to each target, a disturbed (according to Eq. 3) coverage is presented. The best choice for the Attacker is target T3. Hence, the Defender's expected payoff equals -0.72.

$(-0.5) + 0.20 \cdot (1 - 0.6) \cdot (-0.9) + 0.205 \cdot (1 - 0.1) \cdot (-0.8) + 0.205 \cdot (1 - 0.4) \cdot (-0.5) + 0.21 \cdot (1 - 0.2) \cdot (-0.3) = -0.3585.$

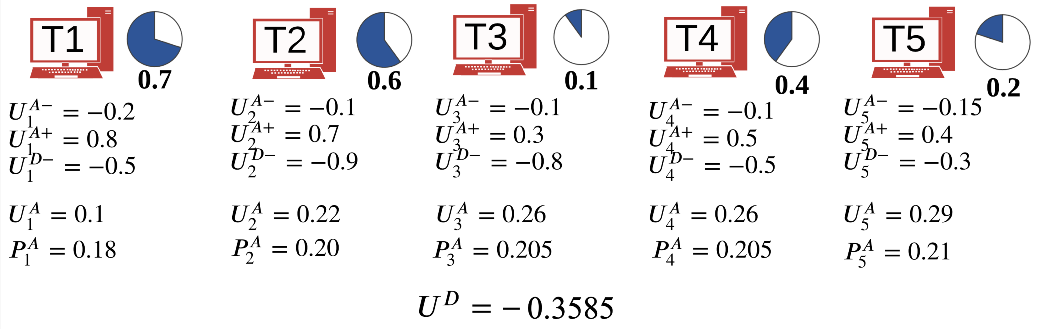


Figure 5: Example game with the assumption that the Attacker adopts the **Quantal Response** decision-making model. Values of attack probabilities P_t^A are calculated according to Eq. 4. In the case of QR, there is no single target to be attacked. The Defender's expected payoff is calculated based on the attack probability distribution and equals -0.3585.

3. Related work

From the perspective of this study, methods of solving SSGs published in the literature can be roughly divided into two groups: those which assume perfectly rational players (let's call them *traditional approaches*) and the ones which consider boundedly rational Attackers.

3.1. Traditional approaches

3.1.1. Exact methods

There are two main categories of solution methods for SSGs: exact and approximate. Exact methods utilize Mixed-Integer Linear Programming (MILP) [27] to formulate SSGs as optimization problems with linear constraints and compute optimal strategies using specialized software. The primary disadvantage of MILP methods is their exponential time and memory complexity.

One notable example of an exact method is BC2015 [28], which extends the DOBBS algorithm [27] (designed for solving a simpler class of one-step Security Games) to extensive-form games [7]. BC2015 transforms an extensive-form game into its equivalent sequence-form representation, reducing the size of the linear program from exponential (as in DOBBS) to linear with respect to the game tree size.

Another popular exact method is C2016 [29]. Like BC2015, C2016 also utilizes MILP, but instead of directly computing the Stackelberg Equilibrium, it utilizes the Stackelberg Extensive-Form Correlated Equilibrium (SEFCE). In SEFCE, the Defender can send signals to the Attacker, who must follow them in his/her choice of strategy. C2016 uses a linear program to compute

SEFCE and then modifies it by iteratively restricting the signals the Defender can send to the Attacker, ultimately converging to SE. The experimental evaluation presented in [29] shows that C2016 is a more time-efficient method than BC2015.

Subsequent work [30] shows that in some cases the complexity of SEFCE can be reduced to polynomial time, which has been adopted in several new algorithms for computing the optimal correlated strategy. However, since this algorithm can only be applied to some special game subclasses, in this paper C2016 is applied to calculate the reference optimal solutions in the experimental evaluation of DNESG.

3.1.2. Approximate methods

Approximate methods provide a viable alternative to exact methods and are able to calculate close-to-optimal solutions much faster, especially for larger games that are beyond the capabilities of exact methods. An example of an approximate method is CBK2018 [31], which is a time-optimized MILP algorithm.

[32] adopt the concept of finite state machines (FSM), whose states represent players' actions, to model SSG strategies. Using FMS can reduce the complexity of computing near-optimal SE by considering only a fraction of strategy space.

A new line of approximate methods based on Monte Carlo Tree Search (MCTS) [33] combined with UCT sampling [34] of the game tree has been recently proposed in [35]. Subsequent works have been developed along two main approaches. The first one, the Mixed-UCT method [36], uses imperfect-information UCT to sample gradually stronger Attacker in order to derive an approximation of the optimal Defender's mixed strategy. The second one, O2UCT [37], combines sampling the Attacker's strategy space with calculating the respective best Defender's strategy for which the sampled Attacker's strategy is the optimal response.

Another approach (EASG [15, 38]) bases on evolutionary computation techniques. EASG maintains a population of candidate Defender's strategies and applies specifically designed mutation and crossover operators. The method is designed as a general framework that can be adapted to various types of SSGs. EASG is also used as a base of our previous neuroevolutionary approach NESG [16] as well as the method proposed in this paper. Section 4 describes EASG in more detail.

In addition to the above general methods, there are also certain heuristic approaches specific to particular SSG formulations, such as [39, 40] which are designed for games on a plane, i.e. in continuous space or [41, 42, 43] which address SSGs with signaling (the Defender can send some signals/alerts to deter or warn the Attacker).

The underlying feature of the above-mentioned methods is the assumption about perfectly rational players who make optimal decisions. In the domain of security, however, particularly with regard to Attackers (such as hackers, thieves, or terrorists), it is unlikely that their choices are always optimal. Taking this decision-imperfection into account can be advantageous for the Defender and result in his/her higher payoffs. Recognizing the decision-making biases of the Attacker potentially allows the Defender to exploit this knowledge by means of adjusting his/her mixed strategy.

3.2. Bounded rationality modeling

The concept of BR has been studied in the literature mainly in the context of single-step games. One of the first BR implementations in SSGs is the COBRA method [9], which modifies the DOBSS MILP [27] to address the Attacker's behavior with ε-optimality model (the Attacker chooses a strategy randomly from a subset of strategies which are worse from the optimal strategy by at most ε). A similar approach is taken by Yang et al. [10, 23] who propose BR models relying on PT and QR, respectively, and demonstrate their suitability in SSGs through experiments involving human players. The SHARP system [11] considers certain game-related aspects, such as past performance and similarity of game conditions, in repeated SSGs played against human adversaries. The MATCH method [12] optimizes the Defender's strategy against a worst-case outcome within some error bound, assuming certain deviations from the Attacker's optimal strategy. Another approach, BRQR [44], proposed by Yang et al., refers to the idea of QR. The method is further improved in the SU-BRQR system [45], which introduces a subjective utility function for the Attacker, with parameters tuned in experiments involving human players. QR is also used to model bounded rationality in the context of the optimal defense resources allocation in power systems [46]. [47] introduces the nested QR adversary model and points shortcomings of the QR model related to the assumption that all choices are made independently.

All the above-mentioned works are implementations of BR models in MILP formulations of single-step SSGs. To the best of our knowledge, the only solution which incorporates BR (in particular AT) into sequential SSGs has been recently proposed in [13].

All existing BR solution methods assume a particular model of BR, while in reality, the Defender usually has no knowledge about the BR model the Attacker follows. Furthermore, there is no single, objectively the best BR concept that can be universally used. To address these limitations, we propose a neuroevolutionary system that (1) does not require any assumptions about a particular BR model of the Attacker, (2) is able to learn the Attacker's decision-making model based on past data, and (3) optimizes the Defender's strategy accordingly.

4. Neuroevolutionary approach

The neuroevolutionary method presented in this paper is an extension of our previous model – NESG, published in the conference paper [16]. The main innovation is a new, simplified and more effective approach to Defender's payoff assessment. In NESG, a neural network is used to directly assess the chromosome fitness in the evaluation/selection phase. Technically, the network solves a regression problem to approximate the expected Defender's payoff in case a strategy provided in the input is played. The network architecture is presented in Figure 6. Due to the inherent difficulty of this regression task, for bigger games the mean absolute error of the network was relatively high and sometimes exceeded the difference between the best and the worst Defender's strategies [16].

In order to address this issue and improve the overall efficacy of NESG, instead of solving the regression problem, we propose to solve a series of binary classification problems, each of them determining which of the two input strategies yields higher payoff for the Defender. This modified approach results in visibly higher network accuracy and consequently leads to the Defender's strategies with higher payoffs.

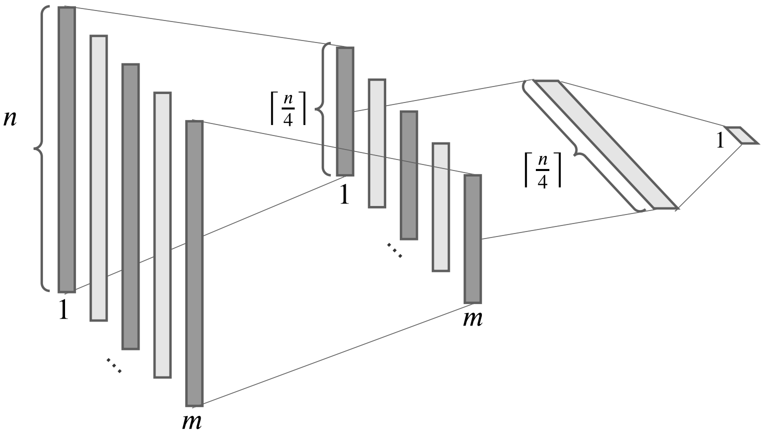


Figure 6: Architecture of strategies evaluation neural network used in NESG method [16].

4.1. Strategy comparison neural network

A strategy comparison neural network (SCNN) is presented in Figure 7. SCNN is a multilayer perceptron with $2nm$ input neurons, 3 hidden layers, and one output neuron. SCNN takes two Defender's strategies π_1^D and π_2^D as the input. Since π_1^D and π_2^D are mixed strategies and their length (number of contained pure strategies) varies, we encoded them as target coverages: respectively $c_1^s(t)$ and $c_2^s(t)$, i.e. probabilities that at least one Defender's unit is allocated to target t in time step s. Thus, the neural network input vector has the following form:

$$\begin{aligned} v_{in} = \big(&c_1^1(t_1), c_1^1(t_2), \ldots, c_1^1(t_n),\ c_1^2(t_1), c_1^2(t_2), \ldots, c_1^2(t_n), \ldots, \\ &c_1^m(t_1), c_1^m(t_2), \ldots, c_1^m(t_n), \\ &c_2^1(t_1), c_2^1(t_2), \ldots, c_2^1(t_n),\ c_2^2(t_1), c_2^2(t_2), \ldots, c_2^2(t_n), \ldots, \\ &c_2^m(t_1), c_2^m(t_2), \ldots, c_2^m(t_n)\big) \end{aligned}$$

The input signals are processed to the first hidden layer (1hl), separately for each strategy and each time step coverage. In the 1hl, target coverages from all steps are compressed four-fold, each of them to the size $\lceil \frac{n}{4} \rceil$. Then in the 2hl, compressed signals from all m time steps are combined into one vector of the size $\lceil \frac{n}{4} \rceil$, still separately for each of the two input strategies. 3hl combines two $\lceil \frac{n}{4} \rceil$ representations of the input strategies into a common representation of size $\lceil \frac{n}{4} \rceil$. Finally, a single unit with a hyperbolic tangent activation function provides the network's output. The output value < 0 is interpreted as $U^D(\pi_1^D) > U^D(\pi_2^D)$ (the first strategy yields a better Defender's payoff). Otherwise, $U^D(\pi_1^D) < U^D(\pi_2^D)$ (the second strategy provides a higher payoff for the Defender).

4.2. Duel-based NeuroEvolutionary system for Security Games (DNESG)

Figure 8 presents an overview of DNESG. The system does not include direct chromosome evaluation. Instead, the selection is performed by means of multiple binary tournaments using SCNN.

4.2.1. Defender's strategy representation

Each individual in the population represents one Defender's mixed strategy: $\pi^D = \{(\sigma_i^D, p_i)\}$, $\sum_i p_i = 1$, $i \in \{1, \ldots, l\}$ where l is the number of pure

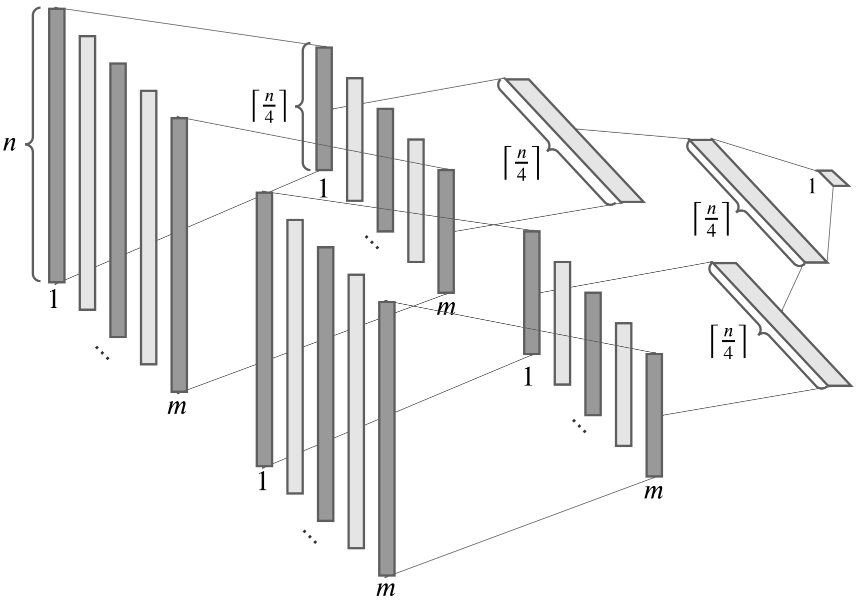


Figure 7: Architecture of strategy comparison neural network. SCNN takes two Defender's strategies, each of them in the form of the target coverages in consecutive steps (from 1 to m), as its input. Input signals from two strategies are processed separately until the last hidden layer, in which they are combined. The output neuron indicates the winning strategy.

strategies composing π^D. Each pure strategy (σ_i^D) defines units allocation to targets in consecutive time steps: $\sigma^D = \{a_{us}\}$, where $a_{us} \in T$ is target allocation for unit u in time step s, $u \in \{1, \ldots, k\}, s \in \{1, \ldots, m\}$.

4.2.2. Initial population

The initial population is composed of pure strategies, i.e. $l = 1$, $p_1 = 1$ for all chromosomes. Each pure strategy is generated randomly, i.e. for each Defender's unit and each time step a random target is chosen. However, in a given time step no two units are assigned to the same target – drawing a target is performed from the set of non-covered targets.

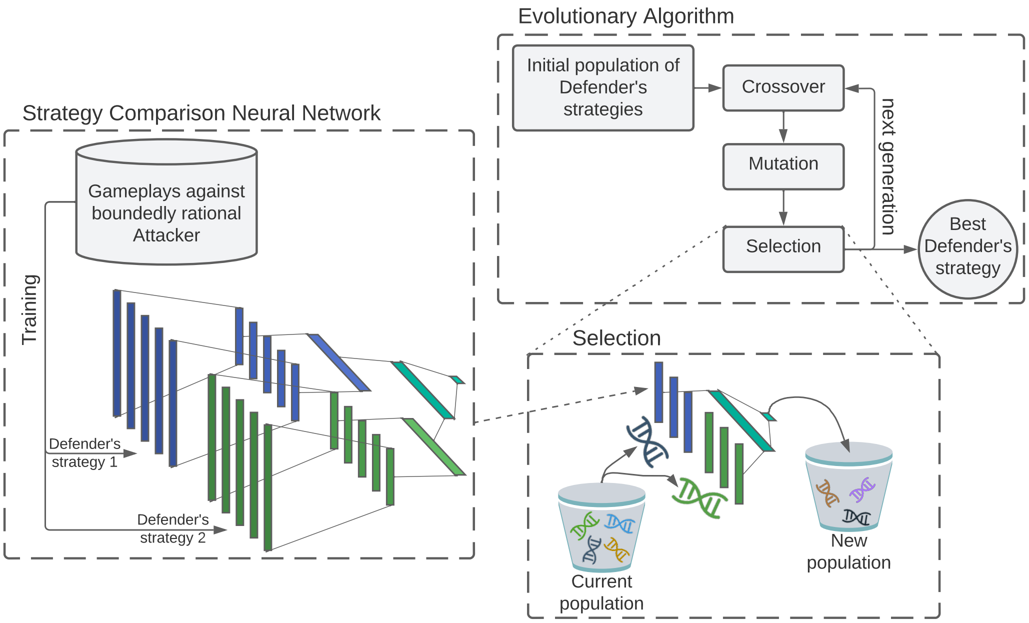


Figure 8: DNESG system overview.

4.2.3. Crossover

Crossover combines pure strategies from two randomly paired individuals (parents). Each individual is selected for the crossover with some fixed probability c_r (*crossover rate*).

In the first step of the crossover operation, a subset of $c_r \cdot p_{size}$ individuals is randomly selected from the population, where p_{size} is the population size. Then, individuals from this subset are randomly paired (in the case of an odd number of individuals, a randomly chosen one is omitted). From each pair of chromosomes, a new offspring individual is created in the following way. All pure strategies from the parent chromosomes are merged into one mixed strategy with their probabilities halved. Then each pure strategy σ_i in this newly-created chromosome, besides the one with the highest probability, is removed with probability $(1 - p_i)^2$ (the lower the probability of a strategy, the higher its chance of being deleted). Next, probabilities of the remaining pure strategies are normalized to sum up to 1. Please refer to [15] for the rationale behind the above crossover formulation.

4.2.4. Mutation

Mutation introduces random perturbation to a mixed strategy represented by a chromosome. Each pure strategy included in the mixed strategy is modified with probability 0.5. The modification changes the target assignment for a randomly selected Defender's unit in a randomly chosen time step. A new target is chosen from the subset of all targets that are not covered in a given time step.

The mutation operator is applied to each individual independently with probability m_r (*mutation rate*).

4.2.5. Selection

Typically, in evolutionary methods, the selection procedure creates a new population based on the chromosome's fitness value computed in the evaluation procedure. In DNESG, evaluation is performed indirectly with no need for the individual's fitness computation.

Let's denote by $SCNN : {\Pi^D}^2 \to \Pi^D$ a function implemented by the neural network. $SCNN$ takes two Defender's strategies $\pi_1, \pi_2 \in \Pi^D$ and returns the one of them that yields higher Defender's payoff (according to the network's assessment), i.e. $SCNN(\pi_1, \pi_2) = \arg\max_{\pi \in \{\pi_1, \pi_2\}} U^D(\pi)$.

The selection procedure conducts a series of binary tournaments, and as a result the new population (for the next generation) is gradually built, until p_{size} individuals are reached. In each tournament, two individuals are randomly selected from the *pool* ($\mathcal{P}$) and compared by the SCNN model described in Section 4.1. Strategies encoded in the two tournament participants form the SCNN input. SCNN output determines which of these two individuals should be promoted to the next generation. Sampling of the tournament participants is performed with return, i.e. selected individuals can participate in subsequent tournaments. In particular, it is possible for the same individual to be promoted multiple times – its multiple copies are added to the new generation. The *pool* $\mathcal{P}$ is composed of all individuals from the current generation, extended by all chromosomes that have undergone crossover and/or mutation. Algorithm 1 presents a pseudocode of the selection procedure.

Algorithm 1: Selection procedure. A new generation population $\mathcal{P}_{new}$ of p_{size} individuals based on the current *pool* $\mathcal{P}$ is created.

```
SelectionTournament ($\mathcal{P}$)
    $\mathcal{P}_{new} \leftarrow Elite(\mathcal{P})$ // new population
    while $|\mathcal{P}_{new}| < p_{size}$ do
        $\pi_1^D, \pi_2^D \leftarrow$ getRandom($\mathcal{P}$) // 2 random strategies from $\mathcal{P}$
        $\mathcal{P}_{new} \leftarrow \mathcal{P}_{new} \cup SCNN(\pi_1^D, \pi_2^D)$
    return $\mathcal{P}_{new}$
```

Note that in our previous neuroevolutionary approach (NESG) there was an additional parameter named selection pressure p_s. With probability p_s, the individual with the higher fitness value was copied to the new generation. Otherwise, the lower-fitted one was promoted. In DNESG there is no need for such a parameter, since promoting lower-fitted individuals is still possible with a certain chance as a result of SCNN incorrect classification.

Please observe that at the beginning of the selection procedure in Algorithm 1, two best individuals are unconditionally transferred to the next generation. This technique (called *elitism*) preserves the best solutions from being forgotten in the evolution process. However, without the fitness function values, identifying these two elite individuals is not straightforward. We apply SCNN to address this problem. The procedure of finding elite individuals is presented in Algorithm 2. First, the algorithm samples two random individuals from the population, SCNN determines which of them represents a better strategy, and they are marked as the *best* and *second best*, respectively. Then, for each individual, the algorithm checks (using SCNN) if the currently considered individual is better than the *best* and/or the *second-best* individual found so far. If so, an adequate update is performed. The whole procedure requires at most $2|\mathcal{P}| - 3$ comparisons made by SCNN.

5. Experimental setup

5.1. Benchmark games

We used the same set of 90 benchmark game instances as in the NESG evaluation [16]. For each number of time steps $m \in \{1, 2, 4\}$ and each number of targets $n = 2^i$, $i \in \{2, \ldots, 7\}$ 5 games were created. Payoffs U_t^{D-} and U_t^{A-} were real numbers independently drawn from interval $(-1, 0)$, while U_t^{D+} and U_t^{A+} were sampled from $(0, 1)$. The number of Defender's units was drawn from the interval $[\lfloor \frac{n}{4m} \rfloor, \lceil \frac{3n}{4m} \rceil]$ (independently for each game instance), i.e. at least $\frac{1}{4}$ and at most $\frac{3}{4}$ of the targets could be effectively protected.

Algorithm 2: Elite selection from the *pool* $\mathcal{P}$.

```
Elite ($\mathcal{P}$)
    $\pi_1^D, \pi_2^D \leftarrow$ getRandom($\mathcal{P}$) // 2 random strategies from $\mathcal{P}$
    $\pi_{best}^D \leftarrow \mathrm{SCNN}(\pi_1^D, \pi_2^D)$
    if $\pi_{best}^D = \pi_1^D$ then
        $\pi_{secondBest}^D \leftarrow \pi_2^D$
    else
        $\pi_{secondBest}^D \leftarrow \pi_1^D$
    for $\pi_i^D \in \mathcal{P} \setminus \{\pi_1^D, \pi_2^D\}$ do
        if ($SCNN(\pi_{best}^D, \pi_i^D) = \pi_i^D$) then
            $\pi_{secondBest}^D \leftarrow \pi_{best}^D$
            $\pi_{best}^D \leftarrow \pi_i^D$
        else
            if ($SCNN(\pi_{secondBest}^D, \pi_i^D) = \pi_i^D$) then
                $\pi_{secondBest}^D \leftarrow \pi_i^D$
    return $\{\pi_{best}^D, \pi_{secondBest}^D\}$
```

5.2. Parameterization

In order to make a fair comparison with NESG and EASG methods we follow all evolutionary algorithm parameter values proposed in [16], with no additional parameter tuning. These are $p_{size} = 100$, number of generations $= 1\,000$, $m_r = 0.5$, $c_r = 0.8$, and elite size $= 2$. Also, the same as previously parameter values were assumed in the BR models – AT: $\delta = 0.5$, QR: $\lambda = 0.8$, PT: $\gamma = 0.64, \theta = 2.25, \alpha = \beta = 0.88$.

5.3. SCNN learning

SCNN is a multilayer perceptron with $2mn$, $2m\left\lceil\frac{n}{4}\right\rceil$, $2\left\lceil\frac{n}{4}\right\rceil$, $\left\lceil\frac{n}{4}\right\rceil$ and 1 units in subsequent layers. The network was trained with backpropagation with a minibatch of size 32. Adam optimizer [48] was used with the learning rate set to 0.001 and the exponential decay rates for the moment estimates equaled 0.9 and 0.990. Hyperbolic tangent activation was applied in the output node and rectified linear unit (ReLU) in all other layers.

For each game, training samples for SCNN were generated in the following way. First, 1 000 random Defender's mixed strategies were generated from a set of strategies that included at most 5 pure strategies. Specifically, the number of pure strategies (l) was randomly chosen from the range $\{1, \ldots, 5\}$ and then each of l pure strategies was generated by randomly allocating the Defender's units to targets and drawing the probability of each pure strategy from $(0, 1]$. The probabilities were then normalized to sum to 1. Finally, 50 000 unique pairs of strategies were randomly created from the above-mentioned 1 000 strategies, and they constituted a training set. Please note that a baseline solution (NESG) uses a significantly higher number of input strategies (5 000 vs. 1 000) which makes NESG implementation less feasible in real-world scenarios.

The training data was generated for each BR model and simulated the Attacker's past behavior. It was assumed that the Attacker's decisions were consistent with the respective BR model, and based on this assumption, the optimal Attacker's response was calculated by iterating over all possible strategies. This information was used to determine the exact Defender's payoff for each of the two given input strategies and decide which of them yielded higher payoff. The more profitable strategy was recorded as the expected SCNN output for this training instance.

6. Results

6.1. SCNN accuracy

To evaluate the accuracy of SCNN we generated 50 000 independent test samples (pairs of Defender's strategies) in the same manner as the training data described in the previous section and verified SCNN accuracy in terms of pointing out the better Defender's strategy within the input pair.

Table 1 shows the SCNN accuracy for games with various numbers of targets and time steps. The results are presented for a no-BR variant, which means that no bounded rationality model was applied, as well as three BR models described in Section 3.2: Anchoring Theory (AT), Quantal Response (QR), and Prospect Theory (PT). Each value is an average of 20 runs. Standard deviations are between 0.011 and 0.024.

Obtained results show that the accuracy of SCNN decreases as the number of targets and/or steps increases, which is expected as the games become more complex and the network must process more data – please recall that the size of the SCNN architecture depends on the game size (number of targets and time steps). Moreover, differences in performance can be observed among various BR models. SCNN demonstrates the best accuracy in predicting payoffs for a no-BR model, which is the simplest case. However, distinctly better accuracy is also obtained for QR and AT models compared to PT. Most likely, this can be attributed to the fact that the use of PT affects both game parameters, i.e. the perceived probabilities and the payoffs, whereas the other models only affect one of these two aspects: AT – the probabilities, QR – the payoffs.

6.2. Payoffs comparison

In order to assess the effectiveness of the proposed DNESG method, it was evaluated against 4 other approaches described in Section 3:

- C2016 [29] - generates optimal (exact) solutions without taking into account the Attacker's bounded rationality.
- EASG [15] - is an evolutionary algorithm that generates approximate solutions without incorporating a BR model.
- EASG_BR, where BR $\in$ {AT, QR, PT} - is the EASG method incorporating the respective BR model, i.e. the Attacker's response in the evaluation procedure is calculated assuming a given decision-making model.
- NESG [16] - a neuroevolutionary approach that uses a neural network to directly estimate the Defender's payoff.

For C2016 and EASG, the Defender's strategy was first generated (without considering the Attacker's bounded rationality) and then the associated payoff was calculated under the assumption that the Attacker would not respond optimally but would follow a particular BR model. It is important to note that it is not possible to incorporate BR models directly into MILP solutions (e.g. C2016) as their implementation introduces nonlinear modifications to payoffs and/or probabilities that would require the use of non-linear constraints in MILP, which is beyond the MILP definition.

Table 1: Average accuracy of the surrogate neural network model (SCNN) for various BR models versus the number of targets (n) and time steps (m).

	no-BR			**Anchoring Theory**			**Quantal Response**			**Prospect Theory**		
n \ m	**1**	**2**	**4**	**1**	**2**	**4**	**1**	**2**	**4**	**1**	**2**	**4**
4	0.943	0.938	0.931	0.932	0.931	0.920	0.891	0.888	0.888	0.874	0.873	0.872
8	0.919	0.918	0.915	0.917	0.914	0.907	0.879	0.879	0.878	0.851	0.848	0.845
16	0.890	0.872	0.866	0.877	0.871	0.865	0.848	0.842	0.839	0.770	0.761	0.755
32	0.829	0.820	0.812	0.819	0.813	0.804	0.813	0.807	0.801	0.686	0.671	0.660
64	0.759	0.754	0.748	0.745	0.752	0.737	0.718	0.715	0.703	0.664	0.657	0.635
128	0.702	0.694	0.674	0.699	0.692	0.669	0.671	0.657	0.650	0.635	0.635	0.614

Table 2 presents the average Defender's payoffs in games with varying numbers of time steps. The results demonstrate that DNESG clearly outperforms methods that do not consider bounded rationality (C2016 and EASG). The advantage of DNESG over these approaches increases as the number of targets in the game increases. This suggests that when playing against an Attacker who is not perfectly rational, it is more effective to use the approximate DNESG algorithm than to employ the optimal no-BR strategy generated by C2016.

For a given BR model, the distinction between EASG_BR and DNESG lies in the population evaluation procedure. EASG_BR calculates the exact response of the Attacker using an explicit (known) BR model, and subsequently calculates the Defender's payoff. DNESG, on the other hand, employs SCNN to implicitly estimate the BR model of the Attacker.

EASG_BR assumes possessing knowledge of the Attacker's BR model, which is unrealistic in many practical scenarios. However, thanks to the direct implementation of the BR model, EASG_BR can be considered an oracle method. NESG and DNESG implement more realistic approaches by trying to infer the Attacker's decision-making model from the past data, during the training process. Due to the above reasons, the NESG and DNESG results presented in the table are slightly worse than those of EASG_BR.

In almost all cases the results obtained by DNESG are better than NESG ones, which proves the effectiveness of the modifications proposed in the paper. For each BR model, the advantage of DNESG over NESG is statistically significant according to a 1-tailed paired t-test with a significance level equal to 0.05, and with a normal distribution of data checked by a Shapiro-Wilk test.

Generally speaking, the results demonstrate high effectiveness of DNESG. Close-to-optimal Defender's strategies are obtained repetitively and for various game instances.

While the experimental assessment is very promising, providing any theoretical convergence guarantees of DNESG is a challenging task, and generally, formal rigorous results for (Neuro)Evolutionary Algorithms rarely occur in the literature. On a general note, based on the construction of DNESG operators, it could be proven that multiple applications of the proposed mutation and crossover can, *in principle*, lead to any arbitrary solution. In other words, any mixed strategy (including the optimal one) can be potentially achieved through an application of these operators, regardless of the initial population selection. Similarly, it can be proven that DNESG operators are able to transform any mixed strategy into any other mixed strategy. However, we cannot say anything about the corresponding time requirements.

6.3. Results repeatability

DNESG is a highly non-deterministic method. Creating the initial population, the use of mutation and crossover operators, and the neural network-based selection – all these components contain random factors. Thus, the mere ability to obtain good solutions, discussed in the previous section, is not sufficient for a comprehensive evaluation of the algorithm. An equally important aspect is the ability to reproduce good results.

In order to check the repeatability of DNESG results, for each game, a standard deviation of the Defender's payoffs was computed over 20 runs. The mean standard deviation equaled 0.0023 with the maximal value of 0.0087,

Table 2: Average Defender's payoff comparison for 1, 2, and 4 time-step games with various BR models.

1 step	**Anchoring Theory**					**Quantal Response**					**Prospect Theory**				
n	**C2016**	**EASG**	**$\mathbf{EASG_{AT}}$**	**NESG**	**DNESG**	**C2016**	**EASG**	**$\mathbf{EASG_{QR}}$**	**NESG**	**DNESG**	**C2016**	**EASG**	**$\mathbf{EASG_{PT}}$**	**NESG**	**DNESG**
4	-0.470	-0.472	-0.468	-0.469	-0.468	-0.406	-0.408	-0.404	-0.405	-0.404	-0.419	-0.420	-0.417	-0.418	-0.417
8	-0.456	-0.457	-0.44	-0.44	-0.440	-0.418	-0.422	-0.386	-0.388	-0.387	-0.422	-0.423	-0.407	-0.407	-0.407
16	-0.387	-0.391	-0.371	-0.371	-0.372	-0.377	-0.378	-0.336	-0.338	-0.337	-0.329	-0.335	-0.315	-0.318	-0.316
32	-0.411	-0.412	-0.393	-0.397	-0.394	-0.428	-0.429	-0.39	-0.394	-0.391	-0.397	-0.404	-0.367	-0.37	-0.368
64	-0.579	-0.586	-0.567	-0.568	-0.567	-0.582	-0.584	-0.536	-0.537	-0.536	-0.56	-0.564	-0.483	-0.486	-0.484
128	-0.397	-0.405	-0.369	-0.372	-0.370	-0.578	-0.578	-0.526	-0.529	-0.527	-0.462	-0.463	-0.345	-0.347	-0.346
2 steps	**Anchoring Theory**					**Quantal Response**					**Prospect Theory**				
n	**C2016**	**EASG**	**$\mathbf{EASG_{AT}}$**	**NESG**	**DNESG**	**C2016**	**EASG**	**$\mathbf{EASG_{QR}}$**	**NESG**	**DNESG**	**C2016**	**EASG**	**$\mathbf{EASG_{PT}}$**	**NESG**	**DNESG**
4	-0.566	-0.566	-0.563	-0.564	-0.563	-0.54	-0.541	-0.534	-0.535	-0.534	-0.548	-0.549	-0.547	-0.547	-0.547
8	-0.568	-0.572	-0.553	-0.555	-0.553	-0.526	-0.528	-0.51	-0.512	-0.510	-0.556	-0.556	-0.517	-0.518	-0.517
16	-0.327	-0.331	-0.314	-0.317	-0.314	-0.326	-0.331	-0.301	-0.302	-0.301	-0.326	-0.331	-0.291	-0.294	-0.292
32	-0.499	-0.5	-0.475	-0.479	-0.476	-0.487	-0.487	-0.435	-0.435	-0.435	-0.501	-0.502	-0.454	-0.457	-0.455
64	-0.457	-0.463	-0.427	-0.427	-0.428	-0.421	-0.424	-0.403	-0.408	-0.404	-0.466	-0.471	-0.407	-0.41	-0.408
128	-0.607	-0.614	-0.563	-0.567	-0.565	-0.601	-0.604	-0.54	-0.544	-0.541	-0.593	-0.595	-0.566	-0.571	-0.568
4 steps	**Anchoring Theory**					**Quantal Response**					**Prospect Theory**				
n	**C2016**	**EASG**	**$\mathbf{EASG_{AT}}$**	**NESG**	**DNESG**	**C2016**	**EASG**	**$\mathbf{EASG_{QR}}$**	**NESG**	**DNESG**	**C2016**	**EASG**	**$\mathbf{EASG_{PT}}$**	**NESG**	**DNESG**
4	-0.479	-0.481	-0.478	-0.479	-0.478	-0.487	-0.489	-0.485	-0.486	-0.485	-0.511	-0.512	-0.508	-0.51	-0.508
8	-0.497	-0.5	-0.466	-0.467	-0.466	-0.509	-0.513	-0.455	-0.456	-0.455	-0.517	-0.519	-0.496	-0.499	-0.497
16	-0.545	-0.547	-0.525	-0.525	-0.525	-0.531	-0.534	-0.502	-0.503	-0.502	-0.57	-0.574	-0.535	-0.538	-0.536
32	-0.478	-0.484	-0.46	-0.464	-0.461	-0.5	-0.505	-0.468	-0.47	-0.468	-0.525	-0.531	-0.492	-0.496	-0.494
64	-0.563	-0.568	-0.547	-0.551	-0.549	-0.587	-0.593	-0.553	-0.555	-0.554	-0.6	-0.600	-0.561	-0.563	-0.562
128	-0.531	-0.536	-0.493	-0.497	-0.494	-0.545	-0.549	-0.503	-0.505	-0.503	-0.553	-0.555	-0.512	-0.512	-0.513

which are very low values relative to the Defender payoffs' range.

6.4. Time scalability

Figure 9 compares the time scalability of the algorithms. The computation time for neuroevolutionary approaches (NESG and DNESG) is presented from two perspectives: one – including the time for neural network training, and the other one – without taking into account the training time (only the inference time is considered). Typically, the training is performed beforehand as a separate step and does not impact the decision-making process.

The analysis of Figure 9 indicates that C2016 (MILP-based) algorithm exhibits a significant increase in computation time as the number of targets increases, in contrast to evolutionary methods (EASG, EASG_BR, NESG, and DNESG). NESG and DNESG are faster than EASG and EASG_BR due to their method of calculating the Defender's strategy, which utilizes a neural network instead of iterating over all possible Attacker's strategies and finding the best response (which is the most time-consuming part of EASG [49, 50]).

Computation times of NESG and DNESG are close to each other. Both solutions make $O(p_{size})$ requests to the respective neural network: NESG to evaluate each of p_{size} individuals, DNESG to compare individuals in pairs ($O(p_{size})$) and to perform the elite selection ($O(p_{size})$). Since the network in DNESG is bigger (it takes two Defender's strategies in the input) its training time is slightly higher than the NESG network.

7. Conclusions

In this paper, we propose a novel neuroevolutionary method (DNESG) for calculating the Defender's payoff in Stackelberg Security Games with

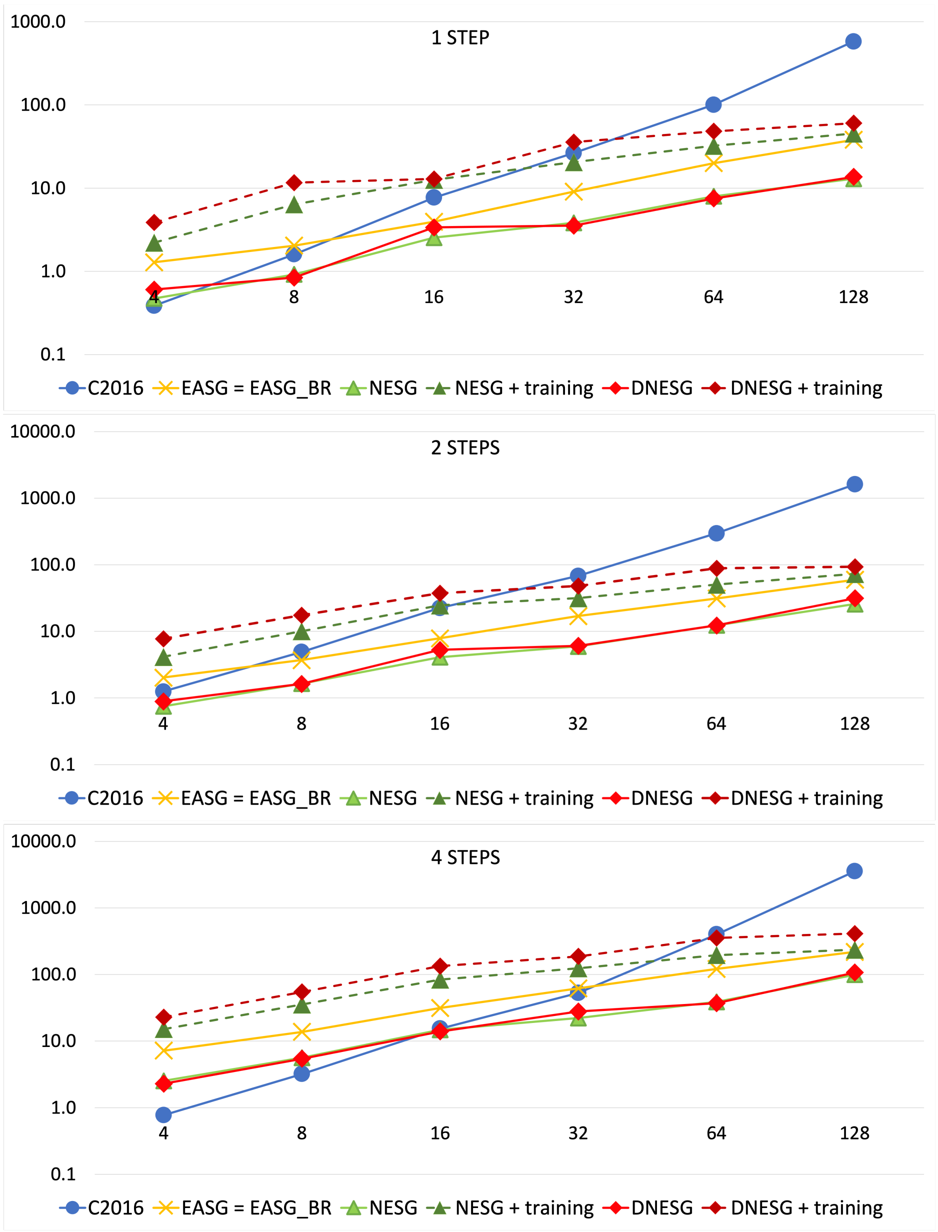


Figure 9: Time scalability of different methods. EASG and EASG_BR are shown as one line since the difference between their computation times is negligible. Dotted lines indicate the time which includes neural network training (for NESG and DNESG).

bounded rationality. The core of the introduced solution is the strategy comparison neural network (SCNN) that can effectively compare two candidate Defender's strategies without having explicit knowledge about the Attacker's payoff distribution or bounded rationality model.

In the security management area, it is often infeasible for the Defenders to have complete information about the Attacker's payoff distribution. Typically, the existing algorithms assume that the Attacker is perfectly rational, which may not be true in practice due to cognitive biases, incorrect perception, or imperfect information [51]. The proposed neuroevolutionary method does not require assuming the perfect rationality of the Attacker, and instead can infer the implicit Attacker's decision-making model through learning from historical data. The method uses this inferred model to evolve highly efficient Defender's strategies.

Experimental evaluation performed on 90 game instances with various characteristics and 3 popular BR models demonstrates the superiority of the proposed approach when playing against Attackers who do not exhibit perfectly rational behavior. DNESG offers high-quality solutions with improved computational scalability.

A novel surrogate neural network model (SCNN), instead of directly approximating the Defender's payoff (like in NESG), pairwise compares the Defender's strategies in a series of duels, which turns out to be a significantly simpler task. A selection phase of the evolutionary process and an elite mechanism are modified accordingly to utilize the outcomes of the series of SCNN duels when creating a new generation.

Thanks to this, the DNESG results are statistically significantly better than the NESG, and training the SCNN model requires a notably smaller amount of historical data (past gameplays).

Stackelberg Security Games have been successfully applied to numerous security domains, for instance the system for scheduling Los Angeles International Airport canine patrols [52], the PROTECT system for randomizing schedules of US Coast Guard's resources in Boston harbour [53], the TRUSTS system for scheduling patrols for fare inspection in Los Angeles Metro system [54], or the PAWS system to prevent poaching and protecting wildlife in Queen Elizabeth National Park in Uganda [55].

Due to knowledge-free design and generic formulation, the proposed DNESG method can be applied to various real-world security scenarios, including the above-mentioned problems. One example of such a scenario (Deep Packet Inspection) is considered in the paper.

A potential limitation of DNESG is the necessity of possessing historical data related to past Attacker's activities, which is required to train the SCNN to approximate the Attacker's behavior scheme (bounded rationality model) for a given problem setup.

Our future plan is to extend DNESG to address more complicated Security Games, e.g. games with a certain degree of the opponent's observability or games with multiple heterogeneous Defenders and/or Attackers [56, 57, 58]. Another direction of planned works is verification of the proposed method in various real-life problems.

References

[1] A. Sinha, F. Fang, B. An, C. Kiekintveld, M. Tambe, Stackelberg Security Games: looking beyond a decade of success, in: Proceedings of the 27th IJCAI Conference, 2018, pp. 5494–5501.

[2] J. Pita, M. Jain, J. Marecki, F. Ordóñez, C. Portway, M. Tambe, C. Western, P. Paruchuri, 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 AAMAS Conference, 2008, pp. 125–132.

[3] F. Fang, P. Stone, M. Tambe, When security games go green: Designing defender strategies to prevent poaching and illegal fishing, in: Proceedings of the 24th IJCAI Conference, 2015, pp. 2589–2595.

[4] H. Guo, Q. Chen, M. Shahidehpour, Q. Xia, C. Kang, Bidding behaviors of GENCOs under bounded rationality with renewable energy, Energy 250 (2022) 123793.

[5] X. Wan, D. Yang, Z. Teng, Blockchain digital technology empowers e-commerce supply chain sustainable value co-creation decision and coordination considering online consumer reviews, Applied Soft Computing 130 (2022) 109662.

[6] A. Sinha, T. H. Nguyen, D. Kar, M. Brown, M. Tambe, A. X. Jiang, From physical security to cybersecurity, Journal of Cybersecurity 1 (1) (2015) 19–35.

[7] V. Conitzer, T. Sandholm, Computing the optimal strategy to commit to, in: Proceedings of the 7th ACM conference on Electronic commerce, 2006, pp. 82–90.

[8] H. A. Simon, Models of Man: Social and Rational, Wiley, 1957.

[9] J. Pita, M. Jain, F. Ordóñez, M. Tambe, S. Kraus, R. Magori-Cohen, M. Tambe, Effective solutions for real-world Stackelberg games: When agents must deal with human uncertainties, Security and Game Theory (2011) 193–212.

[10] R. Yang, C. Kiekintveld, F. Ordóñez, M. Tambe, R. John, Improving resource allocation strategies against human adversaries in security games: An extended study, Artificial Intelligence 195 (195) (2013) 440–469.

[11] D. Kar, F. Fang, F. D. Fave, N. Sintov, M. Tambe, A Game of Thrones: When Human Behavior Models Compete in Repeated Stackelberg Security Games, Proceedings of the 14th AAMAS Conference (2015) 1381–1390.

[12] J. Pita, R. John, R. Maheswaran, M. Tambe, R. Yang, S. Kraus, A robust approach to addressing human adversaries in security games, in: Proceedings of the 11th AAMAS Conference, 2012, pp. 1297–1298.

[13] J. Karwowski, J. Mańdziuk, A. Żychowski, Sequential Stackelberg Games with bounded rationality, Applied Soft Computing 132 (2023) 109846.

[14] R. Gabrys, M. Bilinski, J. Mauger, D. Silva, S. Fugate, Casino Rationale: Countering Attacker Deception in Zero-Sum Stackelberg Security Games

of Bounded Rationality, in: Decision and Game Theory for Security: 13th International Conference, GameSec 2022, Springer, 2023, pp. 23–43.

[15] A. Żychowski, J. Mańdziuk, Evolution of Strategies in Sequential Security Games, in: Proceedings of the 20th AAMAS Conference, 2021, pp. 1434–1442.

[16] A. Żychowski, J. Mańdziuk, Learning attacker's bounded rationality model in security games, in: International Conference on Neural Information Processing (ICONIP), Springer, 2021, pp. 530–539.

[17] M. Klaes, E.-M. Sent, et al., A conceptual history of the emergence of bounded rationality, History of political economy 37 (1) (2005) 27–59.

[18] K. Daniel, Thinking, fast and slow (2011).

[19] A. Tversky, D. Kahneman, Judgment under uncertainty: Heuristics and biases, Science 185 (4157) (1974) 1124–1131.

[20] D. Kahneman, 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.

[21] A. Tversky, D. Kahneman, Advances in prospect theory: Cumulative representation of uncertainty, Journal of Risk and Uncertainty 5 (4) (1992) 297–323.

[22] R. D. McKelvey, T. R. Palfrey, Quantal response equilibria for normal form games, Games and Economic Behavior 10 (1) (1995) 6–38.

[23] R. Yang, F. Ordonez, M. Tambe, Computing optimal strategy against quantal response in security games, in: Proceedings of the 11th AAMAS Conference, 2012, pp. 847–854.

[24] M. Breton, A. Alj, A. Haurie, Sequential stackelberg equilibria in two-person games, Journal of Optimization Theory and Applications 59 (1) (1988) 71–97.

[25] B. Von Stengel, S. Zamir, Leadership with commitment to mixed strategies, Tech. rep., CDAM Research Report (2004).

[26] R. T. El-Maghraby, N. M. Abd Elazim, A. M. Bahaa-Eldin, A survey on deep packet inspection, in: Proceedings of the 23rd ICCES Conference, 2017, pp. 188–197.

[27] P. Paruchuri, J. P. Pearce, J. Marecki, M. Tambe, F. Ordonez, S. Kraus, Playing games for security: an efficient exact algorithm for solving Bayesian Stackelberg games, in: Proceedings of the 7th AAMAS Conference, 2008, pp. 895–902.

[28] B. Bosansky, J. Cermak, Sequence-form algorithm for computing Stackelberg equilibria in extensive-form games, in: Proceedings of the 29th AAAI Conference on Artificial Intelligence, 2015, pp. 805–811.

[29] J. Cermak, B. Bosansky, K. Durkota, V. Lisy, C. Kiekintveld, Using correlated strategies for computing Stackelberg equilibria in extensive-form games, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence, 2016, pp. 439–445.

[30] B. Bošanskỳ, S. Brânzei, K. A. Hansen, T. B. Lund, P. B. Miltersen, Computation of Stackelberg equilibria of finite sequential games, ACM Transactions on Economics and Computation (TEAC) 5 (4) (2017) 1–24.

[31] J. Černỳ, B. Boỳanskỳ, C. Kiekintveld, Incremental strategy generation for Stackelberg equilibria in extensive-form games, in: Proceedings of the 2018 ACM Conference on Economics and Computation, ACM, 2018, pp. 151–168.

[32] J. Černỳ, B. Bosanskỳ, B. An, Finite state machines play extensive-form games, in: Proceedings of the 21st ACM Conference on Economics and Computation, 2020, pp. 509–533.

[33] M. Świechowski, K. Godlewski, B. Sawicki, J. Mańdziuk, Monte Carlo Tree Search: a review of recent modifications and applications, Artificial Intelligence Review 56 (2023) 2497–2562.

[34] L. Kocsis, C. Szepesvári, Bandit based monte-carlo planning, in: European conference on machine learning, Springer, 2006, pp. 282–293.

[35] J. Karwowski, J. Mańdziuk, A new approach to Security Games, in: Proceedings of the International Conference on Artificial Intelligence and Soft Computing (ICAISC'2015), Vol. 9120 of Lecture Notes in Computer Science, Springer International Publishing, 2015, pp. 402–411.

[36] J. Karwowski, J. Mańdziuk, A Monte Carlo Tree Search approach to finding efficient patrolling schemes on graphs, European Journal of Operational Research 277 (1) (2019) 255 – 268.

[37] J. Karwowski, J. Mańdziuk, Double-oracle sampling method for stackelberg equilibrium approximation in general-sum extensive-form games, in: Proceedings of the 34th AAAI Conference on Artificial Intelligence, 2020, pp. 2054–2061.

[38] A. Żychowski, J. Mańdziuk, A generic metaheuristic approach to sequential Security Games, in: Proceedings of the 19th AAMAS Conference, 2020, p. 2089–2091.

[39] X. Wang, B. An, M. Strobel, F. Kong, Catching Captain Jack: Efficient time and space dependent patrols to combat oil-siphoning in international waters, in: Proceedings of the 32th AAAI Conference on Artificial Intelligence, Vol. 32, 2018, pp. 208–215.

[40] J. Karwowski, J. Mańdziuk, A. Żychowski, F. Grajek, B. An, A Memetic Approach for Sequential Security Games on a Plane with Moving Targets, in: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, 2019, pp. 970–977.

[41] E. Bondi, H. Oh, H. Xu, F. Fang, B. Dilkina, M. Tambe, To signal or not to signal: Exploiting uncertain real-time information in signaling games for security and sustainability., in: Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, 2020, pp. 1369–1377.

[42] A. Venugopal, E. Bondi, H. Kamarthi, K. Dholakia, B. Ravindran, M. Tambe, Reinforcement learning for unified allocation and patrolling in signaling games with uncertainty, in: Proceedings of the 20th AAMAS Conference, 2021, pp. 1353–1361.

[43] A. Żychowski, J. Mańdziuk, E. Bondi, A. Venugopal, M. Tambe, B. Ravindran, Evolutionary approach to Security Games with signaling, Proceedings of the 31st IJCAI Conference (2022) 620–627.

[44] R. Yang, C. Kiekintveld, F. Ordonez, M. Tambe, R. John, Improving resource allocation strategy against human adversaries in security games, in: Proceedings of the 22nd IJCAI Conference, 2011, pp. 458–464.

[45] T. H. Nguyen, R. Yang, A. Azaria, S. Kraus, M. Tambe, Analyzing the effectiveness of adversary modeling in security games, in: Proceedings of the 27th AAAI Conference on Artificial Intelligence, 2013, pp. 718–724.

[46] C.-W. Shao, Y.-F. Li, Optimal defense resources allocation for power system based on bounded rationality game theory analysis, IEEE Transactions on Power Systems 36 (5) (2021) 4223–4234.

[47] T. Mai, A. Sinha, Choices are not independent: Stackelberg security games with nested quantal response models, in: Proceedings of the 34th AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 5141–5149.

[48] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[49] A. Żychowski, J. Mańdziuk, Coevolutionary approach to sequential Stackelberg Security Games, in: Proceedings of the 22nd International Conference on Computational Science (ICCS), Springer, 2022, pp. 103–117.

[50] A. Żychowski, J. Mańdziuk, Coevolution of players strategies in security games, Journal of Computational Science 68 (2023) 101980.

[51] J. B. Clempner, Reveling misleading information for defenders and attackers in repeated stackelberg security games, Engineering Applications of Artificial Intelligence 110 (2022) 104703.

[52] M. Jain, J. Tsai, J. Pita, C. Kiekintveld, S. Rathi, M. Tambe, F. Ordóñez, Software assistants for randomized patrol planning for the lax airport police and the federal air marshal service, Interfaces 40 (4) (2010) 267–290.

[53] E. Shieh, B. An, R. Yang, M. Tambe, C. Baldwin, J. DiRenzo, B. Maule, G. Meyer, PROTECT: A deployed game theoretic system to protect the ports of the United States, in: Proceedings of the 11th AAMAS Conference, 2012, pp. 13–20.

[54] Z. Yin, A. X. Jiang, M. P. Johnson, C. Kiekintveld, K. Leyton-Brown, T. Sandholm, M. Tambe, J. P. Sullivan, Trusts: Scheduling randomized patrols for fare inspection in transit systems, in: Proceedings of the Twenty-Fourth Conference on Innovative Applications of Artificial Intelligence, 2012, p. 59.

[55] R. Yang, B. Ford, M. Tambe, A. Lemieux, Adaptive resource allocation for wildlife protection against illegal poachers, in: Proceedings of the 2014 International Conference on Autonomous Agents and Multiagent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 453–460.

[56] J. Lou, A. M. Smith, Y. Vorobeychik, Multidefender security games, IEEE Intelligent Systems 32 (2017) 50–60.

[57] K. Wang, L. Xu, A. Perrault, M. K. Reiter, M. Tambe, Coordinating followers to reach better equilibria: End-to-end gradient descent for stackelberg games, in: Proceedings of the 36th AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 5219–5227.

[58] Z. Cheng, G. Chen, Y. Hong, Single-leader-multiple-followers stackelberg security game with hypergame framework, IEEE Transactions on Information Forensics and Security 17 (2022) 954–969.