Chapter 3
Security Strategies and Multi-Criteria Decision Making

Stefan Rass

3.1 Introduction

The essence of security is defending assets against an adversary that may behave almost arbitrarily. Game theory can help finding optimal strategies against any possible behavior, provided that the attacker stays within a known action space. This is the typical domain and case of security risk management, where a set of threats is identified, against which a uniformly best defense is sought. In game-theoretic terms, the threat list corresponds to an action space, and the best defense against that list is a *security strategy*. This chapter discusses how such strategies can be computed for single and multiple protection goals, even when the effects of the defense actions are nondeterministic (random). The latter especially admits a treatment of uncertainty in three forms, being about the adversary (form and number), the attacker(s) incentives, and – to a limited extent – also the action space (threat list) itself. Our goal in the following is looking at suitable game-theoretic models and methods to compute best defenses under uncertainty.

In many cases, the information available to a decision maker is uncertain in diverse manners. If at least some information is available, then Bayesian choices [21] appear as the natural way to go, since they aim at minimizing the residual uncertainty given all available information.

What if the information is not uncertain but rather not there at all? In that case, assumptions must be made, but how can we be sure that these are even remotely close to reality? The answer is that this verification problem has no general solution, and cannot be circumvented either. Even the Bayesian approach relies on the specification (assumption) of some a priori distribution (or hyperpriors, in higher

S. Rass (✉)
Universitaet Klagenfurt, Institute of Applied Informatics, Universitaetsstrasse 65-67, 9020 Klagenfurt, Austria
e-mail: stefan.rass@aau.at

© Springer International Publishing AG, part of Springer Nature 2018

S. Rass, S. Schauer (eds.), *Game Theory for Security and Risk Management*,
Static & Dynamic Game Theory: Foundations & Applications,
https://doi.org/10.1007/978-3-319-75268-6_3

order Bayesian methods), which is nothing else but an informed guess about the yet unknown parameters. Alternative to informed guesses that Bayesian decision theory speaks about, minimax decision theory seeks to optimize decisions against any incarnation of the uncertain factors. That is, whatever happens after we made our decision, we have prepared ourselves for the worst and decided for the best in that (worst) case. Such a choice is called a *security strategy*. It comes with the price tag of being presumably pessimistic, and even disregarding much information that would be available perhaps. Indeed, (informed) Bayesian decisions and (uninformed) minimax decisions are closely connected to one another, and both find themselves reflected in the framework of game theory.

In general, any decision being made against a rational opponent or simply against nature (an irrational opponent) can be viewed as a game. The set of actions to choose from makes up the action set AS_1 (synonymously called a strategy set) for the first player. The extent to which that player anticipates what its opponent can do constitutes the action set AS_2 for the second player (in the simplest case). Toward a conventional game, described between two players for simplicity, letting the general case follow soon, let us assume that both players can specify a utility value (payoff) function $u_1, u_2 : AS_1 \times AS_2 \to \mathbb{R}$ that, for each combination of actions, returns a (deterministic) value u_1 for player 1, and u_2 for player 2. This completes the description of a classical game as a triple (N, S, H), in which $N = \{1, 2, \ldots\}$, is the set of players, each of which has an associated action set in $S = \{AS_1, AS_2, \ldots\}$, and another associated utility function in $H = \{u_i : AS_i \times AS_{-i} \to \mathbb{R} : i \in N\}$. The symbol AS_{-i} is the joint action set of i's opponents, being the cartesian product of all A_j for $j \in N \setminus \{i\}$. Typically, theoretical considerations are simplified by assuming a fixed (finite) number of players (in our case, $|N| = 2$), or a fixed (finite) number of strategies. In that case, the game itself is called finite.

Our main interest in this chapter will concern the sources of uncertainty, in light of which security strategies need to be found. So, which would be cases, where any of the three ingredients (or a combination thereof) is uncertain?

- Uncertain number of players: A player surely knows that it is engaging in some sort of competition, but the opponents may not be visible or even classifiable as rational. An example for a game with known players are (classical) auctions, in which all bidders personally face each other (unless the auction happens online). The converse extreme is found in computer security, where attackers are notoriously stealthy, and the exact number of them can fluctuate and is almost always unknown.
- Uncertain strategy spaces: In simple auctions, the game dynamics are clearly defined with precise rules, and a fair auction endows all players with the same strategy spaces, so that there is no uncertainty on this point. Again, computer security is an example of the opposite, where the actions for the defending party are known, while not so for the adversary: from the defender's perspective, there is an unpleasant and even unquantifiable residual chance that the attacker comes up with something completely unexpected. Such an attack is launched "zero days after its discovery," that is, at a time when the defender is still unaware of it. For this reason, this is called a *zero-day exploit*.
- Uncertain utility functions: For the positive extreme, simple auctions allow for an almost exact utility value, which is the same for all players and equal to

the value of the good for which the bidders run. The only uncertainty here is the potentially incorrect pricing of that good, but considering the price itself as the utility, this matter becomes negligible. Computer security games, again, are located at the other end of the spectrum, which becomes evident considering that security has a hardly measurable return on investment. The main purpose of security is to prevent possible losses, rather than to generate revenues. This makes the entire objective of increasing security difficult to argue, and makes things more intricate for decision makers: They cannot choose the action that surely rewards them the most, but must rather find the action that potentially saves them from large cost. In addition, these costs may be difficult to quantify (which is another independent issue).

It is fair to note here that the auctions mentioned above are considered in their simplest form, and the mechanisms and dynamics underneath auctions are sufficiently rich to span all extremes in all three cases above. Our focus in the following, however, will be on computer security, and the related security games. We stress that the coincidence of the naming is what it is here, a mere coincidence. The term "security strategy" has per se nothing to do with (computer) security, and exclusively relates to a minimax choice. That is, a choice against the worst case that could happen. Computer security is only one (among many) natural field of application that we shall use for running illustrations.

A convenient common denominator in the representation of all three of the above cases is obtained by letting the utilities, hereafter also called *payoffs*, be uncertain, or more precisely random variables (r.v.s). With random utility functions, the other two cases become covered:

- Uncertain number of players: If a player faces a varying and unknown number of opponents, it may perceive unexpected payoff fluctuations due to unknown people having taken influence. If the distribution of these fluctuations can be pinned down, then the whole world against which player 1 competes can be viewed as a single player with an unknown action set (physically consisting of many players with individual different capabilities and actions). Here, we assume the adversaries to gain their revenue as a team without negatively impacting each other, so that the payoff for the physical adversary $2,3,\dots,N$ is only coming from the defending player 1. In reality, this may not be the case, which effectively results in the game being not zero-sum (between player 1 and the "team" acting as player 2). As we will show below (rigorously stated by inequality (3.3)), this violation nonetheless leaves the results and properties of security strategies unchanged and intact.
- Uncertain strategy spaces: Those correspond to unknown (undiscovered) parts in the domain on which the utility functions are defined, thus making them appear random to the extent at which the unknown actions are chosen at random.

The framework of stochastic orders, for example, the one put forth in Chapter 2, can be used for maximal flexibility in replacing the utility functions by random variables. We will switch between talking about real-valued or distribution-valued payoffs, whichever is more convenient.

3.2 Security Games with a Single Objective

Noncooperative players usually look for equilibria, that is, a strategy profile $(x_1^*,\dots,x_n^*)$ for all players $i \in N = \{1,2,\dots,n\}$ so that

$$\forall i \in N : u_i(x_i^*, \mathbf{x}_{-i}^*) \geq u_i(x_i, \mathbf{x}_{-i}^*), \qquad \text{for all } x_i \in AS_i \tag{3.1}$$

that is, no player gains by unilaterally deviating from the optimum x_i^*. In absence of any information about the number of opponents or their utility functions, we will need to view the opponent(s) as one big and vague entity, acting as player 2. Since this makes the payoffs necessarily unknown too, we ought to use the only information that is certain, which is our own payoff. In the best case, the opponent's intentions are completely unrelated to our own ones, in which case we can selfishly maximize our own revenue without anyone interfering (or us disturbing someone else). In the worst case, the intentions of us and the other player are opposite, and we both pull at different ends of the same rope. This is a zero-sum competition, in which we put $u_1 = -u_2 =: u$. For that class of two-player games, let the equilibrium be (x^*, y^*), and condition (3.1) boils down to

$$u(x,y^*) \leq u(x^*,y^*) \leq u(x^*,y), \tag{3.2}$$

for all $x,y \in AS_1 \times AS_2$. The existence of either, the zero-sum or general (Nash) equilibrium above is not assured without additional assumptions on the strategy spaces. Usually, we convexify those by turning to randomized strategies from the set (simplices) $S_i := \Delta(AS_i)$ for all $i \in \mathbb{N}$, and redefine the utilities into expectations, again denoted as $U = \mathrm{E}(u(X,Y))$, where the expectation is w.r.t. the distributions of the random strategies X,Y chosen from AS_1, AS_2.

The intuition of a zero-sum game being a valid worst-case model is an almost immediate consequence of (3.2): let $\Gamma = (\{1,2\},\{S_1,S_2\},\{u_1,u_2\})$ be a general game, and call $\Gamma_0 = (\{1,2\},\{S_1,S_2\},\{u_1,-u_1\})$ its associated zero-sum game from player 1's perspective. That is, player 1 does have no clue whatsoever on the payoff and incentives of player 2, yet the action space of both players is common knowledge. So, the best that player 1 can do is engage in Γ_0, while player 2 is actually playing Γ. Call $v = val(\Gamma_0) = \mathrm{E}(u_1(X^*,Y^*))$ the *saddle-point value* of Γ_0 upon equilibrium strategies X^*,Y^* played there. Since player 2 engages in Γ, it probably has a different equilibrium strategy $Y_\Gamma^* \neq Y^*$ and hence unilaterally deviates from (X^*,Y^*), thus increasing player 1's revenue $v \leq u_1(X^*,Y_\Gamma^*)$. Conversely, from player 2's perspective, player 1 deviates from its optimum $X_\Gamma^* \neq X^*$ and hence can only decrease its own revenue upon this. So, the chain of inequalities continues as $u_1(X^*,Y_\Gamma^*) \leq u_1(X_\Gamma^*,Y_\Gamma^*)$, and in total, leads to

$$v = val(\Gamma_0) \leq u_1(X^*,Y), \tag{3.3}$$

for all $Y \in S_2$ that player 2 could follow. That means that whatever incentives player 2 may have, it can never decrease player 1's revenue below $v = val(\Gamma_0)$, as long as player 1 follows its zero-sum optimal equilibrium strategy X^*. This X^* is the sought *security strategy* for player 1, and it can only fail if the strategy space for player 2, which is necessary to compute X^*, is inaccurate. Likewise, the associated zero-sum game for player 1 is called a *security game*.

Remark 3.1. Assuming the action spaces of both players to be mutual knowledge may appear hard and even unjustified in light of zero-day attacks, whose sole characteristic is the action's *unexpectedness*. To a limited extent, taking payoffs as random variables, the tails of the payoff distribution (see Chapter 2 or [18]) admits losses beyond what the actions would be known to imply. The tail region of the loss distribution is then where zero-day exploits would occur. A concrete valuation for zero-day risks is given by [28].

Remark 3.2. Nash equilibria are typically applied in games with full information, but security is essentially a competition with lack of information on both sides. There are several ways to resolve this seeming issue; one is the use of distributions in the payoff structure, another is the transition to stochastic games themselves (such as partially observable Markov decision processes with full or partial observability [29, 12]). Occasionally, the uncertainty is not about what can happen (the system administrator may have quite a decent idea about the entry points in the system, so that the strategy spaces of both players are not too uncertain), but only about what *will* happen. If the strategy spaces are known, yet only the adversary's incentives are uncertain, then Nash equilibria can be applied in this special case. The point of security strategies is the assumption that the adversary's incentives are opposite to that of the defenders (and hence known to the defender). However, the defender does not even need to be sure that s/he engages in a zero-sum competition, since if the game is not zero-sum, then (3.3) will only become a more pessimistic overestimate of the actually suffered loss.

Definition 3.1 (Single-Goal Security Game). Let $\Gamma = (\{1,2\},\{S_1,S_2\},\{u_1,u_2\})$ be a two-player game. The *security game* (for player 1) associated with Γ is the zero-sum game $\Gamma_0 = (\{1,2\},\{S_1,S_2\},\{u_1,-u_1\})$. If Γ_0 admits a Nash equilibrium $(x^*,y^*) \in S_1 \times S_2$, then x^* is called a *security strategy* (for player 1).

Note we assumed nothing about the strategy spaces (not even finiteness), except for them to admit an equilibrium to exist (one suitable condition is compactness of all S_i and continuity of the payoff functions w.r.t. the same topology [7]).

The bound (3.3) that a security strategy implies is generally sharp, as simple examples show:

Example 3.1 ([14]). Consider the two-person nonzero-sum game with payoff structure as in Figure 3.1.

This game has multiple equilibria with values $v_1 \in E_1 = \left\{2, 4, \frac{8}{3}, \frac{18}{7}, \frac{9}{4}, \frac{14}{5}\right\}$ for player 1, and $v_2 \in E_2 = \{2,3\}$ for player 2, with respective strategies and payoffs as listed in Table 3.1. Now, consider the associated security games from player 1, and player 2's perspective (either being the adversary to the other in both cases), having the payoff structures as shown in Figure 3.2.

	Player 2				
	(2,0)	(2,0)	(1,4)	(3,1)	(2,3)
	(1,1)	(2,3)	(2,1)	(2,3)	(4,2)
Player 1	(0,2)	(3,2)	(0,1)	(2,3)	(2,1)
	(0,2)	(4,2)	(1,0)	(0,2)	(1,2)
	(2,3)	(2,1)	(4,3)	(4,1)	(3,0)

Fig. 3.1: Example Nonzero-Sum Game

Security game for player 1:

2	2	1	3	2
1	2	2	2	4
0	3	0	2	2
0	4	1	0	1
2	2	4	4	3

Security game for player 2:

0	0	4	1	3
1	3	1	3	2
2	2	1	3	1
2	2	0	2	2
3	1	3	1	0

Fig. 3.2: Security Games Associated with the bimatrix game in Figure 3.1

Table 3.1: Equilibria (and Security Strategies) for Example 3.1, computed using [2]

#	player 1 equilibrium						player 2 equilibrium					
	x_1^*	x_2^*	x_3^*	x_4^*	x_5^*	u_1^*	y_1^*	y_2^*	y_3^*	y_4^*	y_5^*	u_2^*
1	0	0	0	1	0	2	1/2	1/2	0	0	0	2
2	0	0	0	1	0	4	0	1	0	0	0	2
3	0	0	0	1	0	8/3	0	2/3	0	1/3	0	2
4	0	0	0	1	0	18/7	0	4/7	0	1/7	2/7	2
5	0	0	0	1	0	9/4	1/4	1/2	0	0	1/4	2
6	0	0	0	1	0	14/5	0	3/5	0	0	2/5	2
7	0	0	0	0	1	2	1	0	0	0	0	3
8	0	0	0	0	1	4	0	0	1	0	0	3

(a) Bimatrix Game Equilibrium (Payoffs as in Figure 3.1)

#	player 1 equilibrium						player 2 equilibrium					
	x_1^*	x_2^*	x_3^*	x_4^*	x_5^*	u_1^*	y_1^*	y_2^*	y_3^*	y_4^*	y_5^*	u_2^*
1	2/3	0	0	0	1/3	2	1	0	0	0	0	-2
2	2/3	0	0	0	1/3	2	1/2	1/2	0	0	0	-2
3	0	0	0	0	1	2	1	0	0	0	0	-2
4	0	0	0	0	1	2	1/2	1/2	0	0	0	-2

(b) Security Game Equilibrium for Player 1 (Payoffs as in Figure 3.2)

#	player 1 equilibrium						player 2 equilibrium					
	x_1^*	x_2^*	x_3^*	x_4^*	x_5^*	u_1^*	y_1^*	y_2^*	y_3^*	y_4^*	y_5^*	u_2^*
1	1/6	1/4	0	1/3	1/4	5/3	1/3	1/6	1/6	0	1/3	-5/3

(c) Security Game Equilibrium for Player 2 (Payoffs as in Figure 3.2)

The value of the security game for player 1 is $v_1 = 2 = \min E_1$, so the bound (3.3) is sharp. Switching roles and looking at the security game for player 2, its value is $v_2 = \frac{17}{10} < 2 = \min E_2$, so the bound can be loose as well.

For the sake of simplicity only, let the strategy spaces be finite in the following, so that the optimal randomized actions X^* can be specified as categorical distributions, vectors, $\mathbf{x}^* = (x_1, \dots, x_{|AS_1|})$ and $\mathbf{y}^* = (y_1, \dots, y_{|AS_2|})$. The saddle-point value exists under these assumptions and can be rewritten as $v = \max_{\mathbf{x}} \min_{\mathbf{y}} \mathrm{E}(u(X,Y)) = \min_{\mathbf{x}} \max_{\mathbf{y}} \mathrm{E}(u(X,Y))$ with $X \sim \mathbf{x}, Y \sim \mathbf{y}$. This form reveals why we call the point x^*, y^* at which v is attained with equality a *minimax decision*. For finite games with a payoff matrix $\mathbf{A} = (a_{ij})$, we can write $\mathrm{E}(u(\mathbf{x},\mathbf{y})) = \mathbf{x}^T \mathbf{A} \mathbf{y} = \sum_{i,j} x_i y_j a_{ij}$ and $v = (x^*)^T \mathbf{A} \mathbf{y}^*$. Bayesian decisions can be framed into this by letting y^* be a "least favourable distribution," so that the Bayes' optimal decision becomes the minimax decision. While the details of this are intricate, a more intuitive link is discovered by letting the payoffs be random variables. As in Chapter 2, let us replace $u(\mathbf{x},\mathbf{y})$ by a probability distribution $F(\mathbf{x},\mathbf{y})$ of the random revenue R, so that

$$\Pr(R \leq r) = F(\mathbf{x},\mathbf{y})(r) = \sum_{i,j}^{n} \Pr(R \leq r | i,j) \Pr(i,j) = \sum_{i,j}^{n} x_i y_j F_{ij}(r), \tag{3.4}$$

where $\Pr(i,j)$ is a shorthand for the likelihood of player 1 choosing action i and player 2 taking action j, and F_{ij} is the payoff distribution in the ij-th entry of the payoff matrix in a distribution-valued game. Note the striking similarity of (3.4) with the version for finite (matrix) games mentioned just before.

The beauty of Bayesian decisions lies in their natural capability of improvement upon new information. This corresponds to an a priori distribution π becoming an a posteriori distribution $\pi(\cdot|D)$ upon the data D. The very same concept can be used in games when the payoff is distribution-valued, since there is no conceptual barrier preventing us from calling $\Pr(R \leq r) = F(\mathbf{x},\mathbf{y})(r)$ an a priori distribution, and upon new information D coming in, switching to $\Pr(R \leq r|D) = F(\mathbf{x},\mathbf{y},D)(r)$ as the a posteriori distribution. A Bayesian decision then goes for a minimization of some loss function applied to the posterior. If that loss function is quadratic, then the Bayes decision is the posterior expectation, which is the same as in regular game theory. Other choices, say, the absolute value loss, yield to the median as a replacement for the expectation. Any such design choice can be avoided at all if we resort to stochastic orders to let the distribution itself be the sole payoff (from which any quantity of interest can be computed afterward).

3.3 Multi-Objective Security Games

Decisions are hardly ever made with only one goal in mind of the defender, but the equilibrium definition cannot straightforwardly be extended to vector-valued payoffs, since those are no longer totally ordered. For any two vectors $\mathbf{u} = (u_1, \dots, u_n), \mathbf{v} = (v_1, \dots, v_n) \in \mathbb{R}^n$, we will write $\mathbf{u} \leq \mathbf{v}$ iff $u_i \leq v_i$ for all

$i = 1,2,\ldots,n$. The converse relation in which there is at least one index i for which $u_i \geq v_i$, irrespectively of what the other components do, is denoted as $\mathbf{u} \geq_1 \mathbf{v}$. The relations $\geq$ and (its complement) $\leq_1$ are defined in the obvious way. In replacing $\leq$ by $\leq_1$ in (3.1), we obtain a *Pareto-Nash equilibrium*, in which any unilateral deviation from the equilibrium will decrease the payoff for the respective player for at least one of its goals.

Security games and security strategies can be defined by turning the previous observations made for single-goal security games into requirements, toward an axiomatic definition. In this regard, we will demand the bound (3.3) to hold for each goal (we call this *assurance*), and to be optimal as in Example 3.1 (this will be the *efficiency* axiom) [15].

Definition 3.2 (Multi-Goal Security Strategy (MGSS)). A strategy $\mathbf{x}^* \in S_1$ in two-player game with continuous vector-valued payoff $\mathbf{u}_1 : S_1 \times S_2 \to \mathbb{R}^d$ for $d \geq 1$ for player 1. Let us denote the i-th coordinate function in $\mathbf{u}_1$ as $u_1^{(i)} : S_1 \times S_2 \to \mathbb{R}$. The competition in which player 1 engages is called a MGSS with *assurance* $\mathbf{v} = (v_1,\ldots,v_d)$ if two criteria are met:

1. The assurances are the component-wise guaranteed payoff for player 1, that is, for all goals $i = 1,2,\ldots,d$, we have
$$v_i \leq u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) \qquad \forall \mathbf{y} \in S_2, \tag{3.5}$$
with equality being achieved by at least one choice $\mathbf{y}_i^* \in S_2$.
2. At least one assurance becomes void if player 1 deviates from $\mathbf{x}^*$ by playing $\mathbf{x} \neq \mathbf{x}^*$. In that case, some $\mathbf{y}_0 \in S_2$ exists such that
$$\mathbf{u}_1(\mathbf{x},\mathbf{y}_0) \leq_1 \mathbf{v}. \tag{3.6}$$

Constraint (3.5) can be stated in a more compact, yet weaker, form by saying that

$$\mathbf{v} \leq \mathbf{u}_1(\mathbf{x}^*,\mathbf{y}), \qquad \forall \mathbf{y} \in S_2. \tag{3.7}$$

The idea to assure existence of a MGSS follows similar lines as before: Let player 1 engage in a hypothetical one-against-all competition where each goal for player 1 relates to its own zero-sum game against a hypothetical opponent. The opponents themselves act independently of each other, and each optimizes only a single goal. This leads to the sibling of the associated zero-sum game Γ_0 from above, which we call the *security game* here to distinguish it from Γ_0 (in previous literature [15], the same concept has been coined an "auxiliary game"; we use the new name here for consistency):

Definition 3.3 (Multi-Objective Security Game (MOSG)). Let Γ be a two-person game, in which only the strategy space and payoff function $\mathbf{u}_1 : S_1 \times S_2 \to \mathbb{R}^d$ for player 1 is known. Let the coordinate functions of $\mathbf{u}_1$ be $u_1^{(1)},\ldots,u_1^{(d)}$. The MOSG associated with Γ is the game $\boldsymbol{\Gamma}_0$ composed from the following ingredients:

- A set of $d+1$ players, in which player 0 is the first player in Γ, having d opponents, each of which corresponds to another of the d goals in Γ.

- An (ordered) multiset of $d+1$ strategy sets being $\{S_1, S_2, S_2, \dots, S_2\}$
- A set of payoff functions $H = \{\mathbf{f}_0, f_1, \dots, f_d\}$. Player 0 is the only one with a vector-valued utility $\mathbf{f}_0 = (f_0^{(1)}, f_0^{(2)}, \dots, f_0^{(d)})$, whose j-th coordinate function is determined by its own action and that of the j-th opponent, that is, $f_0^{(j)} := u_1^{(j)}(\mathbf{x}, \mathbf{y}_j)$. Likewise, the j-th opponent has the scalar payoff $f_j := -u_1^{(j)}$, and the same strategy space S_2 as all other opponents.

Definition 3.3 is made to materialize its foregoing intuition in the way of exhibiting each Pareto-Nash equilibrium (as defined above) in the security game to be an MGSS in the original game. The proof is based on the following result:

Lemma 3.1. *Let Γ be as in Definition 3.3, where the strategy spaces for both players are compact, and let $\mathbf{x}^*$ be a MGSS with assurance $\mathbf{v}$, assuming that one exists. Then, no vector $\widetilde{\mathbf{v}} < \mathbf{v}$ is an assurance for $\mathbf{x}^*$.*

Proof (from [15]). Let $\widetilde{\mathbf{v}} < \mathbf{v}$, put $\varepsilon := \min_{1 \le i \le k}\{v_i - \widetilde{v}_i\}$ and observe that $\varepsilon > 0$. The function $\mathbf{u}_1$ is uniformly continuous on $S_1 \times S_2$ (being compact), so a $\delta > 0$ exists with $\|(\mathbf{x},\mathbf{y}) - (\mathbf{x}',\mathbf{y}')\|_\infty < \delta$ implying $\|\mathbf{u}_1(\mathbf{x},\mathbf{y}) - \mathbf{u}_1(\mathbf{x}',\mathbf{y}')\|_\infty < \frac{\varepsilon}{2}$.

Consider the mapping $\mathbf{u}_\mathbf{y} : S_1 \to \mathbb{R}^k, \mathbf{u}_\mathbf{y}(\mathbf{x}) := \mathbf{u}_1(\mathbf{x},\mathbf{y})$, which is as well uniformly continuous on S_1 by the same argument. So, $\|(\mathbf{x}^*,\mathbf{y}) - (\mathbf{x}',\mathbf{y})\|_\infty = \|\mathbf{x}^* - \mathbf{x}'\|_\infty < \delta$ implies $\|\mathbf{u}_\mathbf{y}(\mathbf{x}^*) - \mathbf{u}_\mathbf{y}(\mathbf{x}')\|_\infty = \max_{1 \le i \le k}\left|u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) - u_1^{(i)}(\mathbf{x}',\mathbf{y})\right| < \frac{\varepsilon}{2} \quad \forall \mathbf{y} \in S_2$. It follows that $\left|u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) - u_1^{(i)}(\mathbf{x}',\mathbf{y})\right| < \frac{\varepsilon}{2}$ for $i = 1,\dots,k$ and all $\mathbf{y} \in S_2$, and consequently $\max_{\mathbf{y} \in S_2}\left|u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) - u_1^{(i)}(\mathbf{x}',\mathbf{y})\right| < \frac{\varepsilon}{2}$. Now, selecting any $\mathbf{x}' \neq \mathbf{x}^*$ within an δ-neighborhood of $\mathbf{x}^*$, we end up asserting $u_1^{(i)}(\mathbf{x}',\mathbf{y}) \ge u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) - \frac{\varepsilon}{2}$ for every i and $\mathbf{y} \in S_2$.

Using $u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) \ge v_i$, we can continue by saying that $u_1^{(i)}(\mathbf{x}',\mathbf{y}) \ge v_i - \frac{\varepsilon}{2} > v_i - \varepsilon$. By definition of ε, we have $v_i - \widetilde{v}_i \ge \varepsilon$, so that $u_1^{(i)}(\mathbf{x}',\mathbf{y}) > \widetilde{v}_i$ for all i, contradicting (3.6) if $\widetilde{\mathbf{v}}$ were to be a valid assurance vector. □

Theorem 3.1. *Let Γ be as in Lemma 3.1. The vector $\mathbf{x}^*$ constitutes a MGSS with assurance $\mathbf{v}$ for player 1 in the game Γ, if and only if it is a Pareto-Nash equilibrium strategy for player 0 in the MOSG $\mathbf{\Gamma}_0$ according to Definition 3.3.*

Proof (from [15]). Throughout the proof, we will put a bar on top of components, that is, payoff functions, that belong to the security game $\mathbf{\Gamma}_0$, to distinguish these from their counterparts in Γ (showing no horizontal bar accent).

("$\Leftarrow$") Let $\mathbf{s}^* := (\mathbf{s}_0^*, \mathbf{s}_1^*, \dots, \mathbf{s}_d^*)$ be a Pareto-Nash equilibrium in $\mathbf{\Gamma}_0$, and set the assurances to

$$v_i := u_1^{(i)}(\mathbf{s}_0^*, \mathbf{s}_i^*) \qquad \text{for all } i = 1, 2, \dots, d. \tag{3.8}$$

We prove that $\mathbf{s}_0^*$ is a MGSS with assurance $\mathbf{v}$. Consider the i-th opponent's point of view. By construction (Definition 3.3), we have his utility independent of the other player's deviations. So regardless if any other opponent deviates, as long as player 0 (his sole rival) plays $\mathbf{s}_0^*$, his strategy $\mathbf{s}_i^*$ is (Pareto-)optimal (notice that his payoff is scalar), thus

$$\begin{aligned}-u_1^{(i)}(\mathbf{s}_0^*,\mathbf{s}_i) &= \overline{u}_i(\mathbf{s}_0^*,\mathbf{s}_1^*,\ldots,\mathbf{s}_{i-1}^*,\mathbf{s}_i,\mathbf{s}_{i+1}^*,\ldots,\mathbf{s}_d^*)\\ &\leq \overline{u}_i(\mathbf{s}_0^*,\mathbf{s}_1^*,\ldots,\mathbf{s}_{i-1}^*,\mathbf{s}_i^*,\mathbf{s}_{i+1}^*,\ldots,\mathbf{s}_d^*) = -u_1^{(i)}(\mathbf{s}_0^*,\mathbf{s}_i^*) = -v_i\end{aligned}$$

for every $\mathbf{s}_i \in S_2$. As this holds for every $i = 1,\ldots,d$, we can conclude $\mathbf{v} \leq \mathbf{u}_1(\mathbf{s}_0^*,\mathbf{s}_2)$ for all $\mathbf{s}_2 \in S_2$. Thus, the first part of Definition 3.2 is verified, since the average outcome of the game cannot undercut its minimum. On the other hand, from player 0's point of view, his strategy $\mathbf{s}_0^*$ is as well Pareto-optimal, that is, by playing $\mathbf{s}_0 \neq \mathbf{s}_0^*$, he ends up with

$$u_1^{(j)}(\mathbf{s}_0,\mathbf{s}_j^*) = \overline{u}_0^{(j)}(\mathbf{s}_0,\mathbf{s}_1^*,\ldots,\mathbf{s}_d^*) \leq \overline{u}_0^{(j)}(\mathbf{s}_0^*,\ldots,\mathbf{s}_d^*) = u_1^{(j)}(\mathbf{s}_0^*,\mathbf{s}_j^*) = v_j$$

for at least one component j, and condition (3.6) of Definition 3.2 is verified.

("$\Rightarrow$") Put $I := \{1,2,\ldots,d\}$. Let $\mathbf{x}^*$ be a MGSS with assurance $\mathbf{v}$. Let $i \in I$ be arbitrary, and assume that $v_i > \min_{\mathbf{y}\in S_2} u_1^{(i)}(\mathbf{x}^*,\mathbf{y})$. In the light of condition (3.7), this is impossible, for otherwise the i-th opponent could play a strategy $\mathbf{y}_i'$ to enforce an outcome $u_1^{(i)}(\mathbf{x}^*,\mathbf{y}_i') = \min_{\mathbf{y}\in S_2} u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) < v_i$, invalidating $\mathbf{v}$ as the assured outcome. The strategy $\mathbf{y}_i'$ necessarily exists because $u_1^{(i)}$ is continuous. Since Definition 3.2(assurance) implies $\min_{\mathbf{y}\in S_2} u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) \leq v_i$ and strict inequality has been ruled out, we must have equality to the minimum and some $\mathbf{y}_i^*$ exists such that

$$v_i = u_1^{(i)}(\mathbf{x}^*,\mathbf{y}_i^*) = \min_{\mathbf{y}\in S_2} u_1^{(i)}(\mathbf{x}^*,\mathbf{y}) = \max_{\mathbf{y}\in S_2} \underbrace{-u_1^{(i)}(\mathbf{x}^*,\mathbf{y})}_{=\overline{u}_i(\mathbf{x}^*,\mathbf{y})} = \overline{u}_i(\mathbf{x}^*,\mathbf{y}_i^*).$$

Therefore, $\mathbf{y}_i^*$ must be an optimal strategy for the i-th opponent if player 0 acts according to $\mathbf{x}^*$. Put $\mathbf{y}^* := (\mathbf{y}_1^*,\ldots,\mathbf{y}_d^*)$. Assume the existence of some MGSS $\mathbf{x}' \neq \mathbf{x}^*$ with uniformly better assurance $\mathbf{v}' > \mathbf{v}$. Then, $\mathbf{v} \leq \mathbf{v}' \leq \mathbf{u}_1(\mathbf{x}',\mathbf{y})$ for all $\mathbf{y} \in S_2$, because (3.7) applies to $\mathbf{x}'$. Take any $\mathbf{x}'' \in S_1$ with $\mathbf{x}'' \neq \mathbf{x}'$. We distinguish two cases: if $\mathbf{x}'' \neq \mathbf{x}^*$, then property (3.6) implies that there is an index j and some $\mathbf{y}$ such that $u_1^{(j)}(\mathbf{x}'',\mathbf{y}) \leq v_j \leq v_j'$. If $\mathbf{x}'' = \mathbf{x}^*$, then by the above argument, we can just use $\mathbf{y}_j^*$ to assert that $u_1^{(j)}(\mathbf{x}'',\mathbf{y}_j^*) = \underbrace{u_1^{(j)}(\mathbf{x}^*,\mathbf{y}_j^*)}_{=v_j} \leq v_j'$ for any index j, thus verifying (3.6). It follows that $\mathbf{v} < \mathbf{v}'$ is as well an assurance for $\mathbf{x}'$, contradicting Lemma 3.1. Hence, there is no $\mathbf{x}'$ for which the assurance $\mathbf{v} = \operatorname{argmin}_{\mathbf{x}\in S_1} \mathbf{u}_1(\mathbf{x},\mathbf{y})$ (in Pareto's sense) with $\mathbf{y}_i = \operatorname{argmin}_{\mathbf{y}\in S_2} u_1^{(i)}(\mathbf{x},\mathbf{y})$ is better than for $\mathbf{x}^*$ in Pareto's sense, proving that the profile $(\mathbf{x}^*,\mathbf{y}^*)$ is a Pareto-Nash equilibrium of $\mathbf{\Gamma}_0$. □

Theorem 3.2 ([10]). *Let $\Gamma = (N,S,H)$ be a Multi-Objective Game (MOG), where each $AS_i \in S$ is convex and compact, and each $\mathbf{u}_i \in H$ is continuous. For each player $i \in N$, let every individual payoff $u_i^{(j)}(s_i,\mathbf{s}_{-i})$ for $1 \leq j \leq r_i$ be a concave function of*

s_i on AS_i, whenever the remaining values $\mathbf{s}_{-i}$ are fixed. Then, Γ has a Pareto-Nash equilibrium.

The existence of MGSS is assured under the usual conditions, for example, finiteness of the game (which reproves a known result of [1] by a simple application of Theorems 3.1 and 3.2):

Corollary 1 (Existence of MGSS in matrix games). *Every finite MOSG has a MGSS in mixed strategies.*

Observe that Definition 3.2 is axiomatic and not limited to finite games or games with a finite number of players. In that sense, the characterization Theorem 3.1 can be combined with other existence conditions for (normal) Nash equilibria to extend the existence of MGSS to various other classes of games.

The proof of Theorem 3.2 is "constructive" in the sense of equating the set of Pareto-Nash equilibria to the set of Nash equilibria in a scalarized version of the MOG. Specifically, [10] prescribe the following steps to find a Pareto-Nash equilibrium in a MOG $\mathbf{\Gamma}$, in which there are n players, the i-th of which having a set of r_i goals to optimize:

1. Fix an arbitrary set of real numbers $\alpha_{11}, \alpha_{12}, \ldots, \alpha_{1r_1}, \alpha_{21}, \ldots, \alpha_{2r_2}, \ldots, \alpha_{n1}, \ldots, \alpha_{nr_n}$ that satisfy condition (3.9):

$$\left.\begin{array}{ll} \sum_{k=1}^{r_i} \alpha_{ik} = 1 & \text{for } i = 1,2,\ldots,n, \text{ and} \\ \alpha_{ik} > 0 & \text{for } k = 1,2,\ldots,r_i \text{ and } i = 1,2,\ldots,n. \end{array}\right\} \tag{3.9}$$

2. Form a (scalar) game $\Gamma_s = (N, S, H')$ with $H' = \{f_1, \ldots, f_n\}$ and

$$f_i = \sum_{k=1}^{r_i} \alpha_{ik} u_i^{(k)}. \tag{3.10}$$

3. Find a Nash-equilibrium $\mathbf{x}^* = (x_1^*, \ldots, x_n^*)$ in Γ_s, which is then a Pareto-Nash equilibrium in $\mathbf{\Gamma}$.

Notice that the Nash equilibria found by the above algorithm depend on the particular choice of weights. Indeed, the full set of equilibria is given as the union of all equilibria over all admissible choices of α's in (3.9) [10].

It is not difficult to verify that by letting player 1 be minimizing (up to here, we implicitly assumed a maximizing first player), all arguments work identically after being rephrased in terms of a total stochastic order such as that from Chapter 2. The results are all the same up to obvious (and purely syntactic) changes toward using $\preceq$ in place of $\leq$. Some qualitative similarities, unfortunately, are lost from this point onward, as shown in Section 3.4.2.1, but can be recovered in an approximate form, as we will discuss in Section 3.4.3.

3.4 Computing Equilibria and Security Strategies

The existence of equilibria in single-goal games is assured by Nash's theorem or generalizations thereof, and methods to compute such equilibria, and hence security strategies, are reviewed below. Computing Pareto-Nash equilibria for getting MGSS (via Theorem 3.1) can, with a little more effort, be reduced to the computation of (regular) Nash equilibria thanks to results in [10]. Thus, it suffices to dig into details about how (normal) Nash equilibria are computed, which we do next.

It must be emphasized that the methods to compute equilibria in the following validly apply without any problems for traditional games over $\mathbb{R}$, but when we switch to distribution-valued games (based on a stochastic order), some methods may no longer work. Conversely, the stochastic $\preceq$-order of Chapter 2 includes $\leq$ as a special case, so that the respective algorithms can, w.l.o.g., be stated in terms of $\preceq$, where the respective version for $\mathbb{R}$ can be obtained by the simple syntactic change of $\preceq$ into $\leq$ everywhere. Still, since there are qualitative differences in the use of $\leq$ or $\preceq$ for the optimization, we let the "problematic" procedures use $\preceq$ to point at the issues with that ordering, letting respective solutions follow.

3.4.1 Solution by Linear Programming

Let the zero-sum games of interest be with finite strategy spaces for both players, so that the payoff structure is a matrix $\mathbf{A} \in \mathbb{R}^{n\times m}$, and consider mixed strategies to ensure the existence of equilibria in all cases. Let these random (mixed) equilibrium strategies X^*, Y^* be characterized by their (categorical) distributions $\mathbf{x}^* \in S_1 = \Delta(AS_1) \subset \mathbb{R}^n, \mathbf{y}^* \in S_2 = \Delta(AS_2) \subset \mathbb{R}^m$. It is not difficult to find the saddle-point value of the game to be $u(\mathbf{x}^*, \mathbf{y}^*) = \max_{\mathbf{x}\in S_1} \min_{\mathbf{y}_2\in S_2} \mathbf{x}^T\mathbf{A}\mathbf{y} = \min_{\mathbf{y}_2\in S_2} \max_{\mathbf{x}\in S_1} \mathbf{x}^T\mathbf{A}\mathbf{y}$ (by strong duality).

For security games, we adopt player 1's perspective, and suppose that player 2 has chosen the (pure and minimizing) strategy $\mathbf{y}$. Then $\mathbf{A}\mathbf{y}$ is a vector, and player 1's objective is the maximization $\max_{\mathbf{x}\in S_1} \mathbf{A}\mathbf{y} = \max_{i=1,\dots,n} \mathbf{e}_i^T\mathbf{A}\mathbf{y}$, where $\mathbf{e}_i$ is the i-th unit vector in $\mathbb{R}^n$. Note that we hereby converted the optimization over a continuum into the much simpler task of choosing the best from a set of finite alternatives (as we previously discussed in Chapter 2). The only constraints added were $\mathbf{x} \geq 0$ and $\mathbf{1}^T\mathbf{x} = 1$, where $\mathbf{1}$ is the vector of all 1's. Substitute $v := \max_{i=1,\dots,n} \mathbf{e}_i^T\mathbf{A}\mathbf{y}$, then the saddle point condition directly translates into the linear program that player 1 needs to solve for finding a security strategy:

$$
\begin{array}{ll}
(P1) & \max\dot{}\,\mathbf{X}\, v \\
 & \text{s.t.} \\
 & v \leq \mathbf{e}_i^T\mathbf{A}\mathbf{x} \quad \forall i = 1,2,\dots,n \\
 & \mathbf{1}^T\mathbf{x} = 1 \\
 & \mathbf{x} \geq 0
\end{array}
\tag{3.11}
$$

This simple formulation admits an exact computation of an equilibrium even in polynomial time for security games as laid out in Section 3.2. For MGSS, the linear programming approach fails because we are dealing with a $(d+1)$-player game, which includes at least three actors in the simplest multi-goal setting. There, we can resort to iterative methods. Similarly, games defined over stochastic orders may not admit the arithmetics needed to solve Equation 3.11, so iterative (learning) methods are the usual method of choice in that cases too (indeed, the stochastic order $\preceq$ from Chapter 2 comes with the full-fledged arithmetic in the hyperreal space, yet lacking an ultrafilter, we have severe difficulties in doing the calculations practically).

3.4.2 Iterative Solutions by Learning

Iterative methods of computing Nash equilibria by online learning (see [8] for a concrete application) let all players start from a suboptimal strategy, and act according to the best of their so-far recorded knowledge to improve their (randomized) strategies. The usual coupled learning method starts from an initial guess for the optimal strategies and utilities, denoted by $\mathbf{x}_{i,0}, \hat{\mathbf{u}}_{i,0}$ for the i-th player. As the (discrete) time $t \in \mathbb{N}$ goes by, both players choose their respective next moves according to some learning rule (cf. [9, Chp.14])

$$\mathbf{x}_{i,t+1} = \Pi_{i,t}(u_{i,t}, \hat{\mathbf{u}}_{i,t}, \mathbf{x}_{i,t}, \lambda_{i,t}, a_{i,t}), \tag{3.12}$$

and update their corresponding utility estimates as

$$\hat{\mathbf{u}}_{i,t+1} = \Sigma_{i,t}(u_{i,t}, \hat{\mathbf{u}}_{i,t}, \mathbf{x}_{i,t}, \lambda_{i,t}, a_{i,t}), \tag{3.13}$$

where $\Pi_{i,t}, \Sigma_{i,t}$ are learning rules that in the most general form depend on the player i, the current time t, the action $a_{i,t}$, and utility $u_{i,t}$ observed for it, as well as the so-far existing estimates for the utility $\hat{\mathbf{u}}_{i,t}$ and (randomized) actions $\mathbf{x}_{i,t}$ at time t. The remaining parameter $\lambda_{i,t}$ covers additional input, for example, a learning rate (to differently weigh recent against past observations) or similar; it will be of no concrete use for us here but is relevant in several other instances of (3.12) and (3.13). We refer the reader to [9] for an in-depth treatment, and confine ourselves to the simplest learning rule called FP. Other such learning regimes can be studied with help of Lyapunov theory applied to the dynamical system that (3.12) and (3.13) induce [9, Chp.14]. Finally, one should bear in mind that the learning model assumes incentive compatibility of the involved players, so that neither player has an incentive to deviate from the learning rules. Deviations thereof that are observable in practice are studied in behaviorial game theory [3], which is outside of our scope here. The broader area treating techniques like this is algorithmic game theory [11, 24] and learning [4, 6].

Let us instantiate (3.12) and (3.13) for two players, let their action history from time 0 to time step $\ell \in \mathbb{N}$ be $x_0, x_1, \ldots, x_\ell \in AS_1$ for player 1, $y_0, y_1, \ldots y_\ell \in AS_2$. Both players alternatingly (or simultaneously) choose their actions to maximize the so-far long-run average, relative to the recorded behavior of the opponent so far. At time t, player 2 takes its move, followed by player 1 who is assumed to observe what its opponent does. Initially, player 1 takes any choice for a pure strategy as a kickoff:

$$\left.\begin{aligned} y_t &= \operatorname{argmax}_{j\in AS_2} \tfrac{1}{t}\textstyle\sum_{\ell=1}^{t} u_2(y_\ell, j) \\ x_{t+1} &= \operatorname{argmax}_{i\in AS_1} \tfrac{1}{t}\textstyle\sum_{\ell=1}^{t} u_1(x_\ell, i) \end{aligned}\right\} \tag{3.14}$$

where u_1, u_2 are the payoffs for players 1 and 2, respectively. The learning regime (3.14) corresponds to $\Pi_{i,t}$ in (3.12), while the arithmetic means appearing in both expressions correspond to the updating of observed revenues in (3.13). It can be shown that FP via (3.14) converges under alternating moves (as stated here) or synchronous moves (where both players choose their next actions at the same time). Various conditions under which (3.14) converges are known, such as the game having a potential, being zero-sum [23] or being general (nonzero-sum) with $|AS_1| = |AS_2| = 2$. In a practical implementation, a careful distinction must be made regarding convergence of the values vs. convergence of the strategies. While the saddle point approximations (3.13) in FP are assured to converge to each other, this is not necessarily happening for the strategies (3.12) as well. Hence, the convergence threshold used to stop the iteration should be imposed on the so-far averaged payoff(s) u_t, say if u_t differs from u_{t+1} only by a residual amount of some a priori chosen $\varepsilon > 0$ in some norm.

Algorithm 1 shows a version of FP for a minimizing first player, implicitly making player maximizing and assuming a zero-sum competition. For generality, the algorithm is formulated over the stochastic order $\preceq$ from Chapter 2 and distribution-valued games here. The $\preceq$-relation orders two random variables X, Y as $X \preceq Y$ if and only if the moment sequence $(\mathrm{E}Y^k)_{k\in\mathbb{N}}$ "diverges faster" than the moment sequence $(\mathrm{E}Y^k)_{k\in\mathbb{N}}$. Practically, it can be shown that the probability mass assigned to the tails of the distributions of X and Y determines the order, so that $X \preceq Y$ holds if and only if extreme events are less likely to happen for X than to occur under Y (see Theorem 2.2 in Chapter 2).

One reason to look at FP in stochastic orders is that finding equilibria in games over those orders is a widely undiscussed issue in the literature, but *could* offer insights into why real players may not always follow a utility-maximization behavior (either because the utility was not accurately modeled, or the order imposed on the utilities is different from the ordering on $\mathbb{R}$; the latter of which is a yet unverified hypothesis and as such a possible subject of research). Also, it pays to formulate the algorithm in more generality, since this version is capable of solving standard games over $\mathbb{R}$ upon a simple tweak that we will describe and justify after the algorithm. Let us first see how and why it works.

In fact, FP over $\preceq$ works exactly as usual, only having the $\leq$-order on $\mathbb{R}$ being replaced by the stochastic $\preceq$, and imposing a pointwise addition of distribution functions where the standard algorithm would only add payoff values. Note that this pointwise addition is crucial here, and perhaps somewhat counterintuitive, since we do not add random variables as usual by convolution. The pointwise addition is due to the sum occurring in the law of total probability (3.4).

Algorithm 1 Fictitious Play

```
Require: an $(n \times m)$-matrix $\mathbf{A}$ of payoff distributions $\mathbf{A} = (F_{ij})$
Ensure: an approximation $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$ of an equilibrium pair $(\mathbf{x}^*, \mathbf{y}^*)$ and two distributions $v_{low}, v_{up}$ so
    that $v_{low} \preceq F(\mathbf{x}^*, \mathbf{y}^*) \preceq v_{up}$. Here, $F(\mathbf{x}^*, \mathbf{y}^*)(r) = \Pr(R \leq r) = \sum_{i,j} F_{ij}(r) \cdot x_i^* \cdot y_j^*$.
 1: initialize $\mathbf{x} \leftarrow \mathbf{0} \in \mathbb{R}^n$, and $\mathbf{y} \leftarrow \mathbf{0} \in \mathbb{R}^m$
 2: $v_{low} \leftarrow$ the $\preceq$-minimum over all column-maxima
 3: $r \leftarrow$the row index giving $v_{low}$
 4: $v_{up} \leftarrow$ the $\preceq$-maximum over all row-minima
 5: $c \leftarrow$ the column index giving $v_{up}$
 6: $\mathbf{u} \leftarrow (F_{1,c}, \ldots, F_{n,c})$
 7: $y_c \leftarrow y_c + 1$                                      ▷ $\mathbf{y} = (y_1, \ldots, y_m)$
 8: $\mathbf{v} \leftarrow \mathbf{0}$                 ▷ initialize $\mathbf{v}$ with $m$ functions that are zero everywhere
 9: for $k = 1, 2, \ldots$ do
10:     $u^* \leftarrow$ the $\preceq$-minimum of $\mathbf{u}$
11:     $r \leftarrow$ the index of $u^*$ in $\mathbf{u}$
12:     $v_{up} \leftarrow$ the $\preceq$-maximum of $\{u^*/k, v_{up}\}$      ▷ pointwise scaling of the distribution $u^*$
13:     $\mathbf{v} \leftarrow \mathbf{v} + (F_{r,1}, \ldots, F_{r,m})$                 ▷ pointwise addition of functions
14:     $x_r \leftarrow x_r + 1$                                 ▷ $\mathbf{x} = (x_1, \ldots, x_r, \ldots, x_n)$
15:     $v_* \leftarrow$ the $\preceq$-maximum of $\mathbf{v}$
16:     $c \leftarrow$ the index of $v_*$ in $\mathbf{v}$
17:     $v_{low} \leftarrow$ the $\preceq$-minimum of $\{v_*/k, v_{low}\}$    ▷ pointwise scaling of the distribution $\mathbf{v}_*$
18:     $\mathbf{u} \leftarrow \mathbf{u} + (F_{1,c}, \ldots, F_{n,c})$                 ▷ pointwise addition of functions
19:     $y_c \leftarrow y_c + 1$                                 ▷ $\mathbf{y} = (y_1, \ldots, y_c, \ldots, y_m)$
20:     exit the loop upon convergence of the strategy vectors (in some norm)
21: end for
22: Normalize $\mathbf{x}, \mathbf{y}$ to unit total sum          ▷ turn $\mathbf{x}, \mathbf{y}$ into probability distributions.
23: return $\tilde{\mathbf{x}} \leftarrow \mathbf{x}$ and $\tilde{\mathbf{y}} \leftarrow \mathbf{y}$                 ▷ $F(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) \approx F(\mathbf{x}^*, \mathbf{y}^*) = (\mathbf{x}^*)^T \mathbf{A} \mathbf{y}^*$
```

How can Algorithm 1 be applied to a normal form game over the reals? Simply by conversion into a game with stochastic payoffs and the same equilibria. The trick is the following: let $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{n \times m}$ be the (real-valued) payoff matrix, where we can assume $a_{ij} \geq 1$ w.l.o.g. Put $a^* := \max\left\{a_{ij}\right\} \geq 1$, and from a_{ij}, define a corresponding Bernoulli random variable $R_{ij} \sim F_{ij}$ with $\Pr(R_{ij} = 1) = \lambda \cdot a_{ij}$ and $\Pr(R_{ij} = 0) = 1 - \Pr(R_{ij} = 1)$. The factor $\lambda > 0$ is the same for all rows and columns. Why does this work? It has been shown in Chapter 2 that $\preceq$ on categorical distributions (and the Bernoulli distribution is one) is essentially a lexicographic order on the probability mass vector, starting from the highest (rightmost) category in descending order. This renders $\Pr(R_{ij} = 1)$ the relevant quantity to choose best actions and add up into a cumulative sum. Since this probability is proportional to a_{ij} by the same factor $\lambda > 0$ for all elements in the payoff structure, the resulting game, when decided upon $\lambda \cdot a_{ij}$, is strategically equivalent to the original game with payoff matrix A. Thus, it shares the same equilibria. For distributions with more categories, the payoffs are merely vectors, and using $\preceq$ as a lexicographic order is equal to playing FP on a “stack” of games. In the first place, the decision about a best reply is made on the matrix containing the probability masses for the highest loss categories. If the decision can be made (lines 12 and 17 in Algorithm 1), then we are done for this iteration. Upon a tie, the probability mass assigned to the second-highest category counts (in the lexicographic order), and the best reply is sought in this (new) matrix.

If there is a tie again, the next level (third highest matrix of category masses) is taken and so on. The process works just the same for continuous distributions, with the only difference of the stack being made for derivatives of increasing order, starting at the 0th derivative (see Lemma 2.2 in Chapter 2). Figure 3.3 illustrates the stack on which FP is done graphically for the case of continuous (in fact, differentiable) payoff densities f_{ij} in the game.

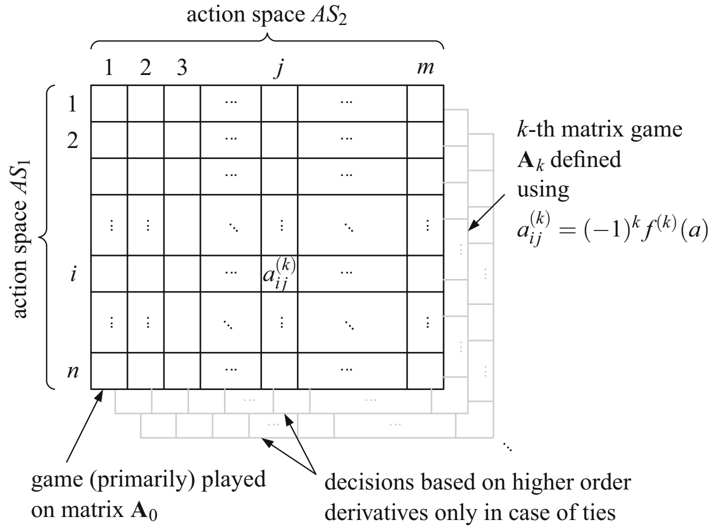

Fig. 3.3: Applying Fictitious Play over a Stochastic Order [18]

The depth of the stack is theoretically unlimited for continuous payoff distributions, thus the algorithm could get stuck within a single iteration during the decision of $\preceq$. In practical applications, we would thus have to fix a finite depth for the stack, and the concrete choice will be discussed later in Section 3.4.2.2.

Example 3.2 ([16]). We construct a 2×2 zero-sum game with payoff matrix $\mathbf{A}$ given in Figure 3.4 and a minimizing player 1.

	Player 2	
Player 1	2	5
	3	1

Fig. 3.4: Example Zero-Sum Game; Payoff Structure $\mathbf{A}$

To use Algorithm 1, the respective payoff distributions representing the game would be (all Bernoulli) $F_{11} = \mathscr{B}er(0.8, 0.2), F_{12} = \mathscr{B}er(0.5, 0.5), F_{21} = \mathscr{B}er(0.7, 0.3)$, and $F_{22} = \mathscr{B}er(0.9, 0.1)$, with $\lambda = 1/10$.

Fictitious play starts from the (arbitrary) choice $\mathbf{x}_0 = (1,0)$ for the first (row) player. This choice causes player 2 to choose the second column in the first time step, to reward player 1 with $u_{1,t=1} = 0.5$. Given player two's history of choices being $\mathbf{y}_{t=1} = (0,1)$, player 1 goes for the second row and chooses $\mathbf{x}_{t=1} = (0,1)$. Player 2 updates its records to make its next choice as a best reply to the so-far observed mixed strategy $\mathbf{x} = (0.5,0.5)$. The switch between the two strategies is essentially due to the game having a circular structure.

It is a straightforward matter to compute the sequence of action choices according to (3.14), verifying them to converge to the equilibrium $\mathbf{x}^* = (0.4,0.6)$, $\mathbf{y}^* = (0.8,0.2)$, and $val(\mathbf{A}) = 2.6$.

To verify this as being a security strategy (for player 1), let us assume that player 2 has different incentives causing it to play $\mathbf{y}' = (0.4,0.6)$ or $\mathbf{y}'' = (0.1,0.9)$. For $\mathbf{y}'$, the payoff for player 1 is $\mathbf{x}^*\mathbf{A}\mathbf{y}' = 2.6$, and $\mathbf{y}''$ gives $\mathbf{x}^*\mathbf{A}\mathbf{y}'' = 2$, both of which are damages $\leq val(\mathbf{A}) = 2.6$. So, in these two cases (at least), player 1 receives no more than the assured maximal damage of $val(\mathbf{A}) = 2.6$. Furthermore, the example shows that worst-case strategies for player 1's opponent are not necessarily unique, and that the bound implied by them can be sharp (as is the case for $\mathbf{y}' \neq \mathbf{y}^*$).

3.4.2.1 Failure of FP in Distribution-Valued Zero-Sum Games

Let us consider what happens if we add uncertainty to the payoffs in Example 3.2. According to the initial discussion, this should cover most interesting cases of uncertainty in the game; however, some qualitative properties such as convergence of FP in zero-sum games are lost upon this transition. We show an example to shed light on the issue and its cause.

Example 3.3 ([16]). Concretely, let each payoff value be uncertain within a certain range, where we model a limited amount of uncertainty by an Epanechnikov kernel ($K(x) := \frac{3}{4}(1-x^2)$ for $|x| \leq 1$ and $K(x) := 0$ otherwise) centered around the respective value x_0. The resulting payoff structure in this game with probability-distribution valued is thus a 2×2 matrix of functions displayed in Figure 3.5.

Note that the game has a circular structure, so that the expected behavior of FP *should* roughly be the following: player 1 choosing the first row will make player 2 choose the second column. In turn, player 1 will go for the second row, which player 2 will reply to by choosing the first row, and so on.

The actual FP algorithm, however, runs elsewhere: let the start be made for player 2 by choosing the $\preceq$-maximum in each row, from which player 1 would select the $\preceq$-minimum. This gives F_{21} as an upper bound to the saddle-point value of this game. Likewise, player 1 will choose the $\preceq$-minimum of the $\preceq$-maxima per column, which gives F_{11} as a lower bound to the saddle-point value. Comparing those to the value 2.6 in Example 3.2, both appear plausible, since F_{11} is centered around 2 and F_{21} is centered around 3, with the value 2.6 lying in between. Moreover, since the upper and lower bounds do not coincide, an equilibrium must be in mixed strategies. Unfortunately, FP will not find it, because the iteration gets stuck at choosing $\mathbf{x}_t = (1,0)$ ultimately for all t, since the losses "accumulate" into $\sum_{j=1}^{k} F_{1y_j}$ for player 1, but we

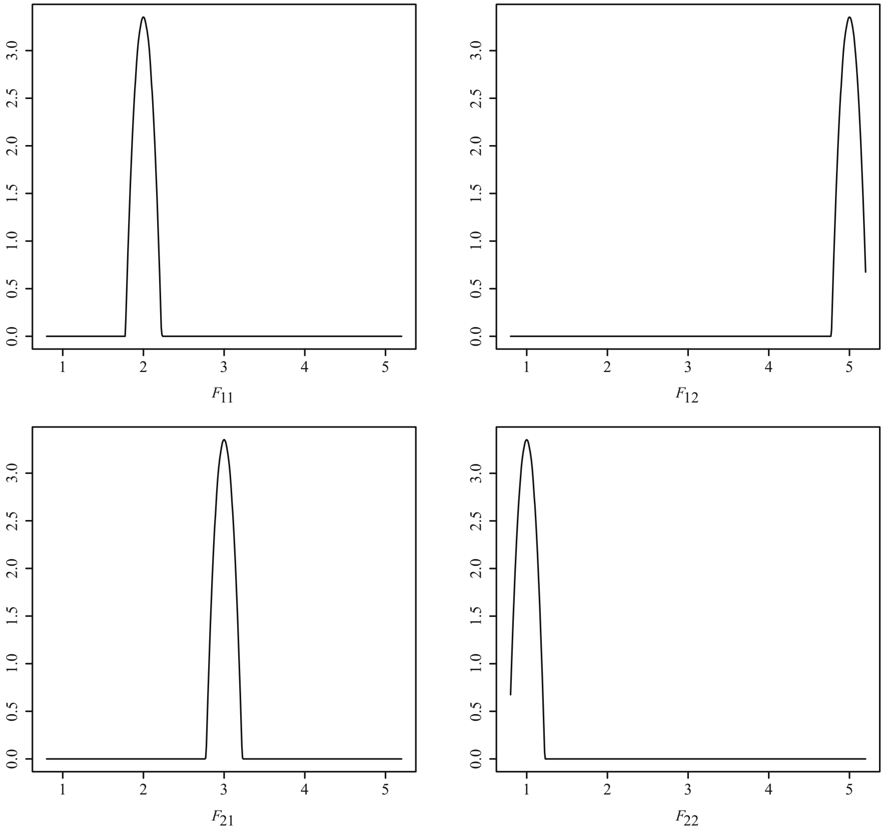

Fig. 3.5: Game from Example 3.2 with uncertainties in the payoffs

have the awkward inequality $F_{11} \preceq \frac{1}{k}u^*$ for all k despite $F_1 1$ and u^* remaining both constant, as Figure 3.6 illustrates! The relation never fails because the tails of the distribution $\frac{1}{k}u^*$ will retain a positive (though decreasing) mass no matter how large k gets; see Figure 3.6 for an illustration. That is, although the losses accumulate, this effect will never justify another choice of strategies, so FP becomes stationary at an incorrect result. Why so? One could think that by the transfer principle [22], the convergence of FP, being a proposition in first-order logic, would equivalently hold in the hyperreals. Indeed, FP *does* converge (as it does classically) by this argument, but for a sequence of *hyperreal* integers, rather than (regular) iterations toward infinity within $\mathbb{N}$. An inspection of the arguments in [23] reveals that the iteration count where convergence occurs is determined by the maximum element in the payoff matrix. Since our distributions are represented by infinite hyperreal numbers, convergence kicks in once the iteration count becomes infinite in the hyperreal sense, which clearly cannot happen in any practical implementation.

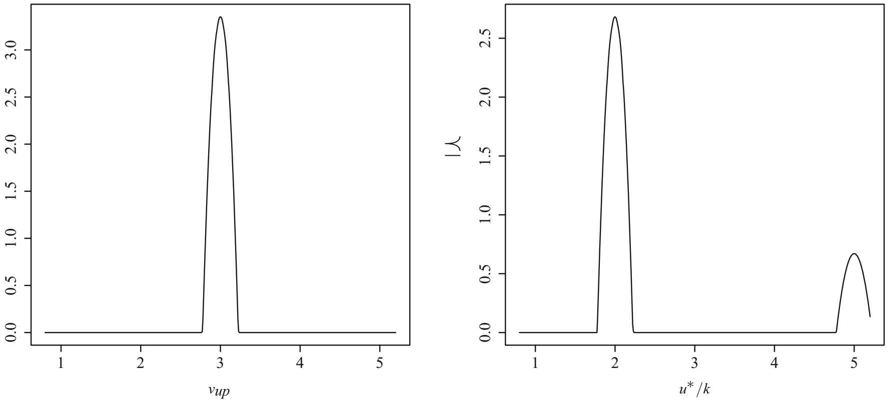

Fig. 3.6: Convergence Failure of FP (situation shown here after $k = 10$ iterations)

The problem outlined in Example 3.3 disappears for distributions with unbounded tails, or if all payoff distributions share the same support $\Omega = [a,b]$ with positive mass assigned in a left neighborhood of b.

3.4.2.2 Restoring Convergence of FP

For the sake of simplicity, let us resort to finite games with continuous payoff distributions, such as the one that caused FP to fail in Section 3.4.2.1. The convergence issue was due to the distribution's tail not reaching out until the point where the stochastic order is decided. Namely, if we consider losses on a bounded scale $[a,b] \subset \mathbb{R}$ (which is a mild and practically handy assumption), the vanishing of the mass located near the end b of the scale along iterations of FP will not be noticed in regions near a (cf. Figures 3.5 and 3.6). To avoid this unpleasant situation, the all relevant distributions must assign strictly positive mass to the entire range $[a,b]$ (so that no "gaps" are near the end or anywhere in the middle of the interval $[a,b]$).

The easiest way of achieving that is convolution by an approximate Dirac mass, say, a Gaussian distribution with small variance, and truncating the resulting density functions. In language of nonparametric statistics, this is nothing else but a standard kernel density estimation (for categorical distributions, a properly discretized Gaussian kernel also works well, but so do more sophisticated methods, e.g., [13, 5], either). In the continuous case, Gaussian kernels come particularly handy for the convolution (see Chapter 2 for the reason), so from here on, we will focus on how and why this also restores convergence of FP. The kernel function is thus $K(x) := \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}x^2\right)$, that is, a humble normal density with zero mean and unit variance.

Let us consider the case of two continuous distributions supported on a compact set $[a,b]$ first, and call them $\tilde{f}, \tilde{g}$. We allow both to vanish on entire intervals within the compact set $[a,b]$. Also, let $K_h : \mathbb{R} \to \mathbb{R}$ be a Gaussian density with variance h that we will use as a mollifier to put $f := \tilde{f} * K_h$ and $g := \tilde{g} * K_h$. It is well known that letting $h \to 0$ makes $f_h \to f$ and $g_h \to g$ in the L^1-norm, and since both are supported on a compact set, the convergence is even uniform. Moreover, since K_h is a C^∞-function, f and g have derivatives of all orders, so that we have $f \preceq g$ (being a shorthand for the relation $X \preceq Y$ between the random variables whose distribution densities are f and g), if and only if the derivatives are lexicographically ordered as $\mathbf{f} = ((-1)^k f^{(k)}(b))_{k\in\mathbb{N}} <_{lex} ((-1)^k g^{(k)}(b))_{k\in\mathbb{N}} = \mathbf{g}$. In the following, let us use the shorthand terms $f_k := (-1)^k f^{(k)}(b)$, and $g_k := (-1)^k g^{(k)}(b)$ to ease notation.

We approximate the infinite sequence by a Taylor polynomial $\hat{f}$ of degree N for f,

$$\hat{f}(x) = f(b) + \sum_{k=1}^{N} \frac{f^{(k)}(b)}{k!}(x-b)^k, \tag{3.15}$$

and do the same for the function g. The choice of the degree N will be discussed later. Let the resulting approximations be $\hat{f}$ and $\hat{g}$. Since there are only finitely many coefficients $f_k = \frac{1}{k!} f^{(k)}(b), g_k = \frac{1}{k!} g^{(k)}(b)$ for $k = 0,1,\ldots,N$ taken to represent the continuously differentiable densities f and g, we can find a (common) bound $M > 0$ so that $-M \le f_i, g_i \le M$ for all $k = 0,1,2,\ldots,M$. Shifting both by the same amount M puts the numbers $f_i + M, g_i + M$ into the interval $[0,2M]$ and leaves their relative ordering unchanged, so that we can consider them as being in excess representation. Fix a precision and round off all numbers $f_i + M, g_i + M$ up to ℓ bits, giving the approximate numbers $\hat{f}_i, \hat{g}_i$ with a roundoff error of $\left|f_i - \hat{f}_i\right|, |g_i - \hat{g}_i| < \varepsilon_M$ for all i. Using $\hat{f}_i, \hat{g}_i$ in the series representation (3.15) for $f^{(i)}(b)$ and $g^{(i)}(b)$, call the resulting approximation polynomials $\hat{f}_\varepsilon$ and $\hat{g}_\varepsilon$. The error from this roundoff is found from (3.15) to be

$$\max_{x\in[a,b]} \left|\hat{f}(x) - \hat{f}_\varepsilon(x)\right| \le \varepsilon_M + \sum_{k=1}^{\infty} \frac{\varepsilon_M}{k!} b^k = \varepsilon_M \cdot e^b,$$

and the same for the error between $\hat{g}$ and $\hat{g}_\varepsilon$. Observe that ε_M can be made as small as we desire by using a larger bitsize ℓ in the numeric representation, so for any $\varepsilon > 0$ there is an ℓ resulting in a roundoff error ε_M so that $\varepsilon_M \cdot e^b < \varepsilon$. So $\hat{f}$ and $\hat{f}_\varepsilon$ can brought together arbitrarily close. Likewise, in choosing N sufficiently large, we can make the difference between f and $\hat{f}$ as small as we wish, so that the cumulative error by the Taylor polynomial and the roundoff errors can be kept under control.

For a number x, let us write $(x)_2$ to mean its binary representation. Using this notation, define the number $y_f := (\hat{f}_0 \hat{f}_1 \hat{f}_2 \ldots \hat{f}_N)_2 \in \mathbb{R}$ and $y_g := (\hat{f}_0 \hat{f}_1 \hat{f}_2 \ldots \hat{f}_N)_2 \in \mathbb{R}$ by a humble string concatenation of the binary excess representations of the (rounded) coefficients in the Taylor polynomials, assuming that they are all represented with the same number of bits. The resulting bitstring is then again interpreted as a real number. Clearly, the information in y_f and y_g can be chosen to represent f and g at arbitrary accuracy, but the numeric order between y_f and y_g is the same as the lexicographic order between $\mathbf{f}$ and $\mathbf{g}$. This in turn equals the $\preceq$-ordering of the original densities f and g.

Wrapping up, we have found real-valued representatives y_f, y_g for f and g so that $y_f \leq y_g$ "implies" $f \preceq g$, where the quotes are a reminder for the relation to be decided on proxima to the original densities.

The goodness of fit is here determined by the number N of coefficients necessary for an accurate approximation by the Taylor polynomial, and the number of bits ℓ in the excess representation (that controls M and hence ε_M). We can thus think of the so-obtained numbers to act as substitutes in games where payoffs are distribution-valued. In other words, we could convert a game with distribution-valued payoffs into a normal game with real-valued payoffs, at the cost of getting only an approximation of the original game, but at any precision that we desire. This equips us with further methods like linear programming (see Section 3.4.1) to solve these games too. Most importantly, in having transformed a game with distribution-valued payoffs into a regular one over $\mathbb{R}$, convergence of fictitious play now follows from standard arguments again [23].

Practically, the number N of required terms in the Taylor approximation, or the number ℓ of bits may become intractably large to be useful any more. Fortunately, however, there is no need to do either a roundoff, excess representation, or binary concatenation into real values, since we can equally well work with vector representations of the series. Then, we can work at machine precision and can compute the derivatives only on demand and up to the index where the decision can be made (exploiting the lexicographic order to be fixed at the time when the first index with a strict relation between f_i and g_i is obtained. Looking at Figure 3.3, we would thus dig only as deep into the stack as we need to make a choice but no deeper than N). Since we expanded the densities around the point b, in whose neighborhood $\preceq$ is determined, the approximation is expectedly accurate in the region relevant for $\preceq$, even for low orders N, though the Taylor polynomial $\hat{f}$ may badly deviate from the real density f when we get far from b. That is, the decision of $\preceq$ based on smoothing and on-demand computation of derivatives is in many cases quite efficient and accurate.

For discrete distributions, matters are considerably simpler, since the smoothing with a density whose support is the entire line $\mathbb{Z}$ of integers (say, by discretizing a Gaussian density to shift their mass from the continuous interval $[n, n+1)$ to the integer n), the support of the distribution extends until the (category/rank) b, and the lexicographic order kicks in again in replacement for $\preceq$. Like before, it is not difficult to assemble the masses together into a single number whose numeric order equals the $\preceq$ ordering, and all theory related to standard games reapplies.

Summing up our arguments (and framing them in more formal terms) leads the following result that relates distribution-valued games to standard (real-valued) games:

Theorem 3.3 (Approximation Theorem [18]). *Let $\Omega \subset [1, \infty)$ be a compact set (finite or continuous). Let $\Delta(\Omega)$ be the set of all distributions for which a density function exists (and is continuous if Ω is continuous). Then, for every $\varepsilon > 0, \delta > 0$ and every zero-sum matrix game $\Gamma_1 = \mathbf{A} \in (\Delta(\Omega))^{n \times m}$ with distribution-valued payoffs in the set, there is another zero-sum matrix game $\Gamma_2 = \mathbf{B} \in \mathbb{R}^{n \times m}$ so that an equilibrium in Γ_2 is an (ε, δ)-approximate equilibrium in Γ_1 in the following sense:*

- *The equilibrium* $(\tilde{\mathbf{x}}^*, \tilde{\mathbf{y}}^*)$ *in* Γ_1 *differs from the equilibrium* $(\mathbf{x}^*, \mathbf{y}^*)$ *in the matrix game represented by* $\mathbf{A}$ *by* $\|(\mathbf{x}^*, \mathbf{y}^*) - (\tilde{\mathbf{x}}^*, \tilde{\mathbf{y}}^*)\|_1 < \varepsilon$*, where the norm is on* $\mathbb{R}^{|AS_1|+|AS_2|}$*,*
- *The saddle point* $val(\mathbf{B}) = \tilde{F}^*$ *differs from the saddle point* $val(\mathbf{A}) = F^*$ *by* $\left\|\tilde{F}^* - F^*\right\|_{L^1} < \delta$*.*

3.4.3 FP for Multi-Goal Security Strategies

For MGSS, it has been shown in [25] that equilibria can be computed by FP for certain one-against-all games, in which a designated player "zero" faces opponents that are acting independently among themselves, but all against player zero. The security game of Definition 3.3 can be modified to fall into this class (cf. [20]).

For a two-player MOG $\mathbf{\Gamma}$, let $\mathbf{\Gamma}_0$ denote its associated security game. Toward enabling fictitious play in $\mathbf{\Gamma}_0$, we need to make it zero-sum. Remember that the defender in $\mathbf{\Gamma}$ has $d \geq 1$ goals to optimize, each corresponding to another distinct opponent in the security game $\mathbf{\Gamma}_0$. From these, we define the payoffs in a one-against-all *compound game*, while making the scalar payoffs vector-valued to achieve the zero-sum property. The payoff for player 0 is left unchanged, but the payoff for the i-th opponent is "vectorized" into

$$\overline{\mathbf{u}}_i = (0, 0, \ldots, 0, -u_1^{(i)}, 0, \ldots, 0), \tag{3.16}$$

without affecting any equilibria in the game (again, the bar accent on top of $\mathbf{u}$ is to mark this and other items with the same accent to belong to the security game $\mathbf{\Gamma}_0$).

To numerically compute an equilibrium in it according to the recipe of [10], we scalarize as follows: to each of player 0's d goals, we assign a weight $\alpha_{01}, \ldots, \alpha_{0d}$ according to (3.9). The scalarization in (3.10) is via

$$\alpha_{ji} := \alpha_{0i} \text{ for } i = 1, 2, \ldots, d \text{ and } j = 1, 2, \ldots, d.$$

With these weights, the payoffs in the scalarized compound game are

- for player 0: $f_0 = \alpha_{01}\overline{\mathbf{u}}_1 + \alpha_{02}\overline{\mathbf{u}}_2 + \cdots + \alpha_{0d}\overline{\mathbf{u}}_d$,
- for the i-th opponent, where $i = 1, 2, \ldots, d$

$$\begin{aligned} f_i &= \alpha_{01} \cdot 0 + \alpha_{02} \cdot 0 + \cdots + \alpha_{0,i-1} \cdot 0 + \alpha_{0i} \cdot (-u_1^{(i)}) + \alpha_{0,i+1} \cdot 0 + \alpha_{0d} \cdot 0 \\ &= -\alpha_{0i} \cdot u_1^{(i)} \end{aligned} \tag{3.17}$$

Concluding the transformation, we obtain a scalar compound game

$$\Gamma_{sc} = (\{0, 1, \ldots, d\}, \{AS_1, AS_2, \ldots, AS_2\}, \{f_0, \ldots, f_d\}) \tag{3.18}$$

from the original two-person MOG Γ with payoffs $u_1^{(1)}, \ldots, u_1^{(d)}$ that can directly be be plugged into expressions (3.16) and (3.17).

Toward a numerical computation of equilibria in Γ_{sc}, we need yet another transformation due to [25]: for the moment, let us consider a general compound game Γ_c as a collection of d two-person games $\Gamma_1, \dots, \Gamma_d$, each of which is played independently between player 0 and one of its d opponents. With Γ_c, we associate a two-person game Γ_{cr} that we call the *reduced game*. The strategy sets and payoffs of player 0 in Γ_{cr} are the same as in Γ_c. Player 2's payoff in the reduced game is given as the *sum* of payoffs of all opponents of player 0 in the compound game. The following result links the convergence of FP in one-against-all games to convergence in their reduced forms.

Lemma 3.2 ([25]). *A fictitious play process approaches equilibrium in a compound game Γ_c if and only if it approaches equilibrium in its reduced game Γ_{cr}.*

For the reduced version Γ_{scr} of the (by (3.16) vectorized) scalarized security game Γ_{scr}, this sum is always zero. Since FP converges in such games [23], we get the final conclusion:

Theorem 3.4 ([20]). *Fictitious play in the scalarized compound game Γ_{sc} defined by* (3.18) *converges to an equilibrium.*

Any Nash equilibrium obtained in Γ_{sc} upon FP in Γ_{src} is by Theorem 3.1, a Pareto-Nash equilibrium in the security game $\mathbf{\Gamma}_0$ and as such a MGSS in the game that we started from. So, Theorem 3.4 induces the following algorithm to compute multi-criteria security strategies according to Definition 3.2:

Given a two-player MOG $\mathbf{\Gamma}$ with d payoffs $u_1^{(1)}, \dots, u_1^{(d)}$ for player 1 (and possibly unknown payoffs for player 2), we obtain a MGSS along the following steps:

1. Assign strictly positive weights $\alpha_{01}, \dots, \alpha_{0d}$ to each goal, and set up the scalarized compound game Γ_{sc} by virtue of expressions (3.18), (3.16), and (3.17). Observe that, as we can choose the weights arbitrarily, these give us a method to *prioritize* different goals.
2. Run the FP Algorithm 1 in Γ_{sc}, stopping when the desirable precision of the equilibrium approximation is reached.
3. The result vector $\mathbf{x}^*$ is directly the sought multi-criteria security strategy, whose assurances are given by the respective expected payoffs of the opponents. In case of matrix games, where the i-th payoff is given by a matrix $\mathbf{A}_i$, the sought assurances are $v_i = (\mathbf{x}^*)^T \mathbf{A}_i \mathbf{y}_i^*$ for $i = 1, 2, \dots, d$, where $\mathbf{y}_1^*, \dots, \mathbf{y}_d^*$ are the other player's equilibrium strategy approximations obtained along FP.

Example 3.4. For ease of presentation and an intuitive validation of the results, let us consider a 2×2 MOG with two goals. The payoff structures, shown in Figure 3.7, are composed from categorical (Bernoulli) distributions. For the example purpose, those cover three possible cases of games: 1) classical games with real-valued outcomes (via the aforementioned representation by Bernoulli random variables), 2) games with random payoffs that are converted into classical games by taking expectations, and 3) the general case of probability-distribution-valued games with categorical distributions compared according to $\preceq$.

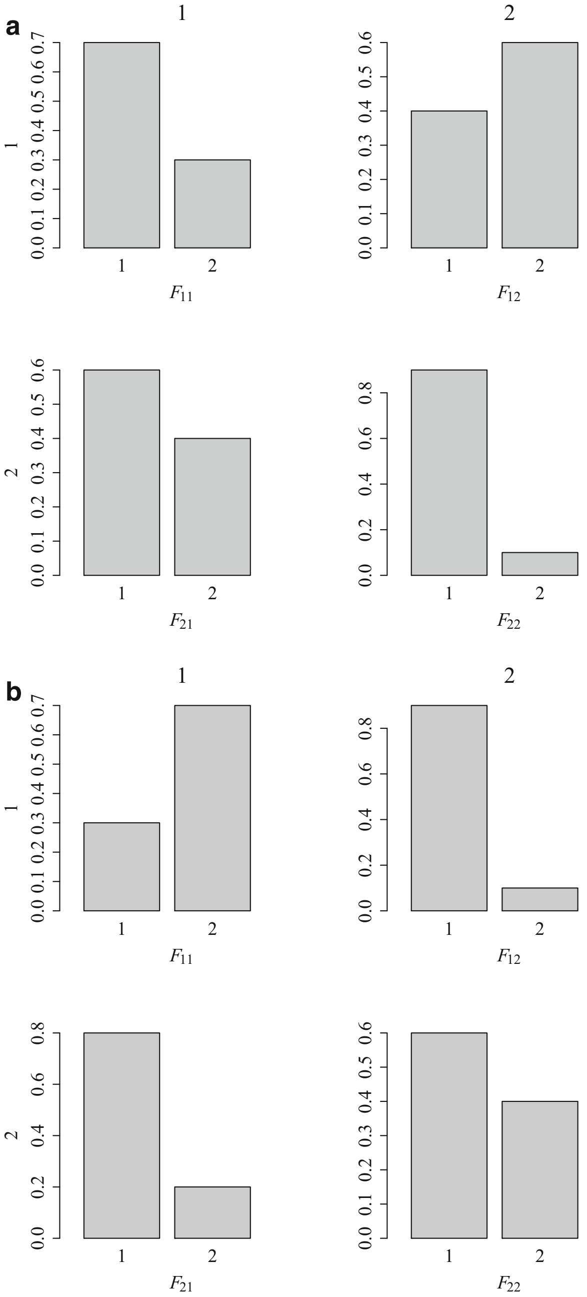

Fig. 3.7: Example Multi-Objective Distribution-Valued 2×2 Game

The following results (and the plots in Figures 3.4 and 3.7) have been obtained with R, version 3.4.4 [27], using the package HyRiM [17], which implements exactly the procedure outlined above with Algorithm 1 at the core. Running Algorithm 1 with equal importance on both goals (i.e., taking the weights $\alpha_{01} = \alpha_{02} = 1/2$) on these games digs up the (approximate) equilibrium $\mathbf{x}^* = (1/4, 3/4)$ and $\mathbf{y}_1^* = (1,0)$ for the first goal, and $\mathbf{y}_2^* = (1/2, 1/2)$ for the second goal. The mixed strategy $\mathbf{x}^*$ is herein the security strategy for player 1, being told the worst-case scenarios for each of his goals to be $\mathbf{y}_1^*$ and $\mathbf{y}_2^*$, respectively. Conditional on the defender playing $\mathbf{x}^*$, the *assurances* are the (Bernoulli) distributions $\mathbf{v}_1 = (0.625, 0.375) = \mathbf{v}_2$ for both goals.

The security strategy is not too sensitive to a change in the goal prioritization. For example, taking $\alpha_{01} = 0.9$ and $\alpha_{02} = 0.1$ to express high importance of the first goal (relative to the second) leaves the security strategy unchanged. Only, the worst-case scenario for the second goal changes into $\mathbf{y}_2^* \approx (0.39, 0.61)$, and its assurance $\mathbf{v}_2$ adapts itself accordingly.

The entire set of equilibria can be discovered by (theoretically) running through all values for the importance weights α_{0i} for $i = 1, 2, \ldots, d$ goals [10]. In a practical setting, one would thus be advised to try different goal priorities in order to find perhaps more plausible equilibria than upon the first try.

3.5 Final Remarks

The assurance offered by a security strategy against whatever behavior of the opponent within its action space is bought at the cost of this being a rather pessimistic approach. As with any minimax decision, this disregards all auxiliary information available to both players, which could improve the decision making. Bayesian decision theory starts from this observation and is developed around the idea of updating loss distributions with incoming data, so as to improve the decisions over time. The same trick, however, can be mounted in game theory, when the game's payoff structures become updated between repetitions. Technically, this makes the games dynamic, but not necessarily stochastic (at least not in the sense of [26]). For distribution-valued games, those can be updated in a Bayesian way, in order to improve the accuracy of the payoff structures. Still, this is not the same as using prior knowledge about the attacker's behavior. However, the same framework allows to integrate that knowledge into the payoff distributions by proper modeling. The details are beyond the scope of this chapter and fall into the domain of general adversary modeling. Hints on how to construct the payoff distributions for several practical cases, however, are subject of Part II of this book. Specifically, the data can be obtained from simulation (Chapters 8, 9, 10, 14, and 15), expert surveys, or other sources. Chapters 8, 14, 15, and 16 report on a practical use of the method, as it is implemented in R [17].

A final remark on security strategies relates to the cost of playing them. Imagine that the equilibrium is mixed and that it prescribes to frequently change configurations or even reset or revert a certain part of the system to some initial state. Frequent such actions may be undesirable and may lead to unreasonably high cost for the defense. Taking into account the cost of playing strategies besides their actual benefits is a matter of multi-objective game theory and can be handled in similar ways as described here. A rigorous treatment of this, however, is beyond the scope of this chapter, but has recently been done in the literature [19].

References

1. Acosta Ortega, F., Rafels, C.: Security Strategies and Equilibria in Multiobjective Matrix Games: Working Papers in Economics
2. Avis, D., Rosenberg, G., Savani, R., von Stengel, B.: Enumeration of nash equilibria for two-player games. Economic Theory (42), 9–37 (2010)
3. Camerer, C.F.: Behavioral game theory: Experiments in strategic interaction. The Roundtable Series in Behavioral Economics. Princeton University Press, s.l. (2011). URL http://gbv.eblib.com/patron/FullRecord.aspx?p=765287
4. Cesa-Bianchi, N., Lugosi, G.: Prediction, learning, and games. Cambridge University Press, Cambridge (2006). URL https://doi.org/10.1017/CBO9780511546921
5. Chesson, P., Lee, C.T.: Families of discrete kernels for modeling dispersal. Theoretical Population Biology pp. 241–256 (2005)
6. Fudenberg, D., Levine, D.K.: The Theory of Learning in Games. MIT Press, London (1998)
7. Fudenberg, D., Tirole, J.: Game Theory. MIT Press, London (1991)
8. Klíma, R., Kiekintveld, C., Lisý, V.: Online Learning Methods for Border Patrol Resource Allocation. In: R. Poovendran, W. Saad (eds.) Decision and Game Theory for Security, *Lecture Notes in Computer Science*, vol. 8840, pp. 340–349. Springer International Publishing, Cham (2014)
9. Lewis, F.L., Liu, D.: Reinforcement Learning and Approximate Dynamic Programming for Feedback Control. John Wiley & Sons, Inc, Hoboken, NJ, USA (2012)
10. Lozovanu, D., Solomon, D., Zelikovsky, A.: Multiobjective Games and Determining Pareto-Nash Equilibria. Buletinul Academiei de Stiinte a Republicii Moldova Matematica **3**(49), 115–122 (2005). ISSN 1024-7696
11. Nisan, N. (ed.): Algorithmic game theory, repr., [nachdr.] edn. Cambridge Univ. Press, Cambridge (2008). URL http://reference-tree.com/book/algorithmic-game-theory?utm_source=gbv&utm_medium=referral&utm_campaign=collaboration
12. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming, *Wiley Series in Probability and Statistics*, vol. v.414. John Wiley & Sons Inc, Hoboken (2009). URL http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=294454

13. Rajagopalan, B., Lall, U.: A Kernel Estimator For Discrete Distributions **4**, 409–426 (1995). Gordon and Breach Science Publishers SA
14. Rass, S.: On Information-Theoretic Security: Contemporary Problems and Solutions. Ph.D. thesis, Klagenfurt University, Institute of Applied Informatics (01.01.2009)
15. Rass, S.: On Game-Theoretic Network Security Provisioning. Springer Journal of Network and Systems Management **21**(1), 47–64 (2013). https://doi.org/10.1007/s10922-012-9229-1. URL http://www.springerlink.com/openurl.asp?genre=article&id=doi:10.1007/s10922-012-9229-1
16. Rass, S.: On Game-Theoretic Risk Management (Part Two) – Algorithms to Compute Nash-Equilibria in Games with Distributions as Payoffs (2015). ArXiv:1511.08591
17. Rass, S., König, S.: R package ‘HyRiM’: Multicriteria risk management using zero-sum games with vector-valued payoffs that are probability distributions (2017). URL https://hyrim.net/software/
18. Rass, S., Konig, S., Schauer, S.: Defending Against Advanced Persistent Threats Using Game-Theory. PLoS ONE **12**(1), e0168,675 (2017). https://doi.org/10.1371/journal.pone.0168675. Journal Article
19. Rass, S., König, S., Schauer, S.: On the cost of game playing: How to control the expenses in mixed strategies. In: Proceedings of the 8th International Conference on Decision and Game Theory for Security (GameSec), LNCS 10575, pp. 494–505. Springer (2017)
20. Rass, S., Rainer, B.: Numerical Computation of Multi-Goal Security Strategies. In: R. Poovendran, W. Saad (eds.) Decision and Game Theory for Security, LNCS 8840, pp. 118–133. Springer (2014)
21. Robert, C.P.: The Bayesian choice. Springer, New York (2001)
22. Robinson, A.: Non-standard Analysis. Princeton Landmarks in Mathematics and Physics. Princeton University Press, Princeton (1996). URL http://gbv.eblib.com/patron/FullRecord.aspx?p=4626045. Luxemburg, W. A. J. (BeteiligteR)
23. Robinson, J.: An iterative method for solving a game. Annals of Mathematics **54**, 296–301 (1951)
24. Roughgarden, T.: Twenty lectures on algorithmic game theory. Cambridge University Press, Cambridge (2016). URL http://dx.doi.org/10.1017/CBO9781316779309
25. Sela, A.: Fictitious play in ‘one-against-all’ multi-player games. Economic Theory **14**(3), 635–651 (1999). URL http://dx.doi.org/10.1007/s001990050345
26. Shapley, L.S.: Stochastic Games. Proceedings of the National Academy of Sciences **39**(10), 1095–1100 (1953). https://doi.org/10.1073/pnas.39.10.1095. URL http://www.pnas.org/content/39/10/1095.short
27. Team, R.D.C.: R: A Language and Environment for Statistical Computing (2016). URL http://www.R-project.org. ISBN 3-900051-07-0

28. Wang, L., Jajodia, S., Singhal, A., Noel, S.: k-zero day safety: Measuring the security risk of networks against unknown attacks. In: D. Hutchison, T. Kanade, J. Kittler, J.M. Kleinberg, F. Mattern, J.C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M.Y. Vardi, G. Weikum, D. Gritzalis, B. Preneel, M. Theoharidou (eds.) Computer Security – ESORICS 2010, *Lecture Notes in Computer Science*, vol. 6345, pp. 573–587. Springer Berlin Heidelberg, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15497_35
29. White, D.J.: Markov decision processes. Wiley, Chichester (1993). URL http://www.loc.gov/catdir/description/wiley033/92001646.html