Toward a Formal Model of Fair Exchange – a Game Theoretic Approach*

Levente Buttyán and Jean-Pierre Hubaux
Institute for computer Communications and Applications
Communication Systems Department
Swiss Federal Institute of Technology
EPFL-DSC-ICA, CH-1015 Lausanne, Switzerland
{Levente.Buttyan, Jean-Pierre.Hubaux}@epfl.ch

4 May 2000

Abstract

A fair exchange protocol is a protocol in which two (or more) mutually suspicious parties exchange their digital items in a way that neither party can gain an advantage over the other. Many fair exchange protocols have been proposed in the academic literature, but they provide rather different types of fairness. The formal comparison of these protocols has remained difficult, mainly because of the lack of a formal framework in which each can be modeled and formal definitions of fairness can be given. In this paper, we introduce game theory as a formal tool to model exchange protocols. We give formal definitions of various types of fairness using standard notions of game theory, and show how the defined fairness types are related to each other. Our results can serve as the foundations of the formal comparison of existing and future fair exchange protocols with respect to fairness.

1 Introduction

Protocols that allow two (or more) mutually suspicious parties to exchange their digital items via communication networks are essential building blocks for electronic commerce services. Examples where such protocols are needed include signing of electronic contracts, certified e-mail delivery, and purchase of network delivered services. In all of these applications, there is an inherent problem, which stems from the fact that the parties distrust each other and may potentially misbehave: a party may end up in a disadvantageous situation. In contract signing, for instance, the party that signs the contract first commits itself without being sure that the other party will also sign the contract. This problem may discourage the parties and hinder otherwise desired transactions. Therefore, an important requirement that exchange protocols should satisfy is *fairness*. Roughly, a fair exchange protocol is a protocol in which the parties can exchange their digital items in a way that neither party can gain an advantage over the other.

*This report is an updated version of EPFL SSC Technical Report No. SSC/1999/039. The earlier version was published in December 1999.

Many scientific papers propose fair exchange protocols (e.g., [Cle89, Ket95, Jak95, Tyg96, DGLW96, ZG96, FR97, ASW97, Syv98, BDM98]). Interestingly enough, besides the details of the proposed protocols, these papers also differ in the interpretation of the concept of fairness informally described above. Examples for various interpretations include the following:

- Early work on fair exchange has resulted in a number of theoretically important fair exchange protocols (e.g., [Cle89]), which are based on gradual secret release schemes. In these protocols, the items of the parties are exchanged in small pieces, typically bit-by-bit. Here, fairness of the protocol means that the computational effort required from the parties to obtain each other's remaining bits is approximately equal at any stage during the execution of the protocol. Clearly, this fairness definition based on equal computational complexity makes sense only if the parties have equal computing power, which is an often unrealistic and undesirable assumption. Therefore, in this paper, we will focus on the other interpretations described below.
- In many of the recent papers that describe practical fair exchange protocols (e.g., [Tyg96, PJ97, BDM98]), fairness is defined (or meant) as some sort of atomicity property, which requires that either both parties obtain the item of the other or none of them gets anything useful.
- Some papers (e.g., [FR97, ASW97]) define fairness as a less demanding property by specifying that the protocol guarantees that correctly behaving parties never suffer a disadvantage.
- Another set of exchange protocols (e.g., [Jak95, Syv98, But99]) provide yet another kind of fairness. When faithfully executed by each party, these protocols ensure a fair outcome, but they do not exclude the possibility that a correctly behaving party suffers a disadvantage if the other party misbehaves. However, in order to ensure that this unfair situation occurs only rarely, the protocols are constructed in a way that misbehavior is uninteresting for each party. This means that, although it may cause some damage to a correctly behaving party, the misbehaving party also loses something or at least does not gain anything (apart from malicious joy) with the misbehavior.

The full understanding of these interpretations and the formal comparison of fair exchange protocols of different types are difficult, mainly because of the lack of a formal framework, in which exchange protocols can be modeled and formal definitions of fairness can be given. In this paper, we introduce game theory [Mor94] to solve this problem. We illustrate the use of game theory in this context in two ways:

1. We model various types of exchange protocols with game trees. A game tree is a directed labeled tree graph that models the possible moves of the protocol participants and the advantages and disadvantages in each possible situation in which the exchange may terminate.
2. We give formal definitions for various types of fairness using standard notions of game theory, and show how the defined fairness types are related to each other.

The outline of the paper is the following. In Section 2, we briefly introduce all the game theoretic notions that we use in this paper. In Section 3, we introduce the idea of modeling exchange protocols with game trees and illustrate it with examples. In Section 4, we propose formal definitions for fairness, and investigate the relationships between these definitions. We discuss some related work in Section 5 and conclude the paper in Section 6.

2 Game theory

Games, such as chess, tic-tac-toe, and many others, can naturally be represented by a labeled directed tree graph. Each vertex of this graph (except for the terminal vertices) is labeled with the name of a *player* and represents a decision point for this player in the course of the game. The choices or possible *moves* at this point are represented by the edges starting from this vertex. The terminal vertices (leaves) of the tree correspond to possible ends of the game. Each leaf is labeled with a tuple of real numbers, which represent the *payoffs* for the players if the game ends in that leaf. The payoff may be negative, in which case it is interpreted as a loss. The starting point of the game is represented by the root of the tree.

A game is said to be *of perfect information* if each player knows at every point in the game the entire previous history of the game. Chess, for instance, is a game of perfect information, but bridge is not (because the cards dealt to the other players are hidden). In the tree representation of games of imperfect information, there always exists at least one set S of vertices, all belonging to a single player P, such that, at a certain point in the game, P knows that she is at one of the vertices in S, but does not know which one. This set is called an *information set for* P. Graphically, we will indicate information sets by dashed boxes that surround the vertices of the sets.

A classical example for a game is The Prisoner's Dilemma. It is described as follows[1]: Two criminals, call them Bonnie and Clyde, are arrested by the police. They are immediately separated so that they cannot communicate in any way. Each is offered the following deal:

- If you confess and implicate the other, then
  - you will serve only 1 year in jail if the other does not confess, or
  - 5 years if the other does confess.
- On the other hand, if you do not confess, then
  - you will serve 10 years if the other confesses, or
  - 2 years if the other does not confess either.

Now, Bonnie and Clyde have to decide, without communicating with each other, to confess or to keep quiet about the crime they have committed. Their decisions will determine the number of years they have to serve in jail.

The game is carefully constructed so that it encourages the criminals to confess. In order to see this, let us imagine how Bonnie may think. Bonnie may assume that Clyde will confess. In this case, Bonnie should also confess, because if she does so, then she has to serve 5 years in jail, while if she keeps quiet, then she gets 10 years. Bonnie may also assume that Clyde will keep quiet. In this case again, Bonnie should confess, because this means 1 year in jail, while if she keeps quiet, she has to serve 2 years. Thus, given that Bonnie cannot communicate and agree on a strategy to follow with Clyde, confessing is the best thing she can do. And the same is true for Clyde.

[1] This presentation of The Prisoner's Dilemma is taken from [Mor94].

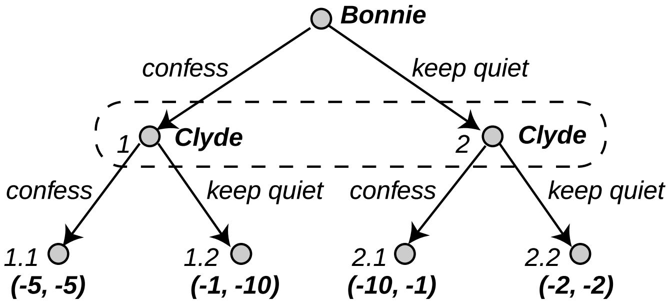

Figure 1: **The Prisoner's Dilemma**

The game tree of The Prisoner's Dilemma is illustrated in Figure 1. In order to facilitate the following discussion, we labeled the edges of the tree with the possible moves of the players and numbered the vertices in addition to the labeling with the players and the payoffs. The game has two players: Bonnie and Clyde. Without loss of generality, we assume that the deal is first offered to Bonnie and then to Clyde without telling him her decision. This means that the game is started by Bonnie, and thus, the root of the tree is labeled by her name. Bonnie has two possible moves: to confess or to keep quiet. These are represented by the two edges starting from the root. Vertex 1 and vertex 2, both of which belong to Clyde, form an information set for Clyde, because he cannot learn Bonnie's decision. Clyde knows that he is at one of these vertices, but he does not know which one. Independently of which vertex he is at, Clyde has two possible moves too: to confess or to keep quiet, and the edges starting from vertex 1 and vertex 2 are labeled accordingly. The game has four possible outcomes, which are represented by the leaves of the tree. Each leaf is labeled with a payoff vector, the first element of which is the payoff for Bonnie and the second element is the payoff for Clyde. In this game, the payoffs are negative and correspond to the number of years that have to be spent in jail. Leaf 1.2, for instance, represents the situation where Bonnie confesses and Clyde keeps quiet, and thus, it is labeled with a payoff of -1 for Bonnie and -10 for Clyde according to the rules of the game.

2.1 Strategies

Informally, a *strategy* for player P is a plan that tells P how to move in any conceivable situation during the game. In order to formally capture this idea, we first introduce the notion of *choice subtrees.* We note that a subtree S of a tree T is a tree, whose vertices form a subset of the vertices of T, whose edges form a subset of the edges of T, whose root is the root of T, and whose leaves form a subset of the leaves of T.

Definition 2.1 *Let T be a game tree and let P be one of the players. A* choice subtree *for P is a subtree S of T, such that for all vertices v in S: (a) if v belongs to P, then exactly one of the children of v is in S; (b) if v does not belong to P, then all the children of v are in S.*

By definition, a choice subtree S for player P always suggests a single move to P at each vertex of S that belongs to P. Furthermore, if P chooses her moves according to S, then the game always remains within S (i.e., it never reaches a vertex, which is not in S). It might seem that a choice subtree for P, indeed, encodes a strategy for P. The problem with choice subtrees is that they do not respect information sets, which means that a choice subtree for P may call for different moves at two vertices, both of which belong to the same information set for P. Since P does not know in which of these vertices she actually is, the choice subtree does not tell her unambiguously how to move in this situation. Therefore, a strategy for P must call for the same move at all vertices that belong to the same information set for P.

Definition 2.2 *Let T be a game tree and let P be one of the players. A* strategy for P *is a choice subtree for P that respects P's information sets.*

The game tree of The Prisoner's Dilemma has four choice subtrees for Clyde, but only two out of the four are strategies for Clyde. These are denoted by $S_{C,c}$ and $S_{C,q}$, and depicted in the lower part of Figure 2. On the other hand, there are two choice subtrees for Bonnie and both are also strategies for Bonnie, because she does not have information sets. The strategies for Bonnie are denoted by $S_{B,c}$ and $S_{B,q}$, and illustrated in the upper part of Figure 2.

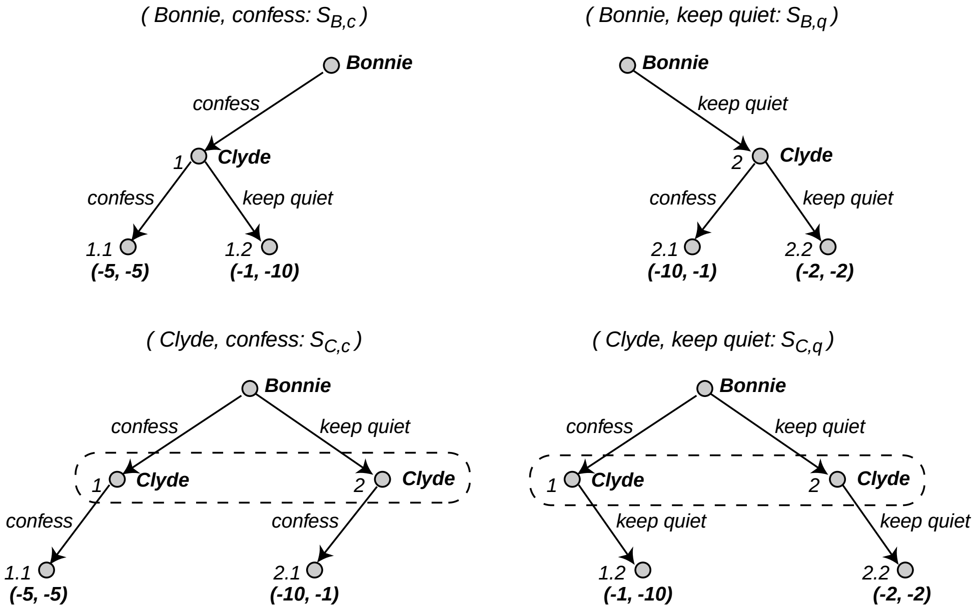

Figure 2: **Strategies for Bonnie and Clyde in The Prisoner's Dilemma**

Let us now consider a game with n players $P_1, P_2, \ldots, P_n$. Let us denote the tree representation of the game by T and the set of all possible strategies for P_i by Σ_i. It is easy to see that for any n-tuple of strategies $(S_1, S_2, \ldots, S_n) \in \Sigma_1 \times \Sigma_2 \times \ldots \times \Sigma_n$ the intersection of the strategies $\cap_{i=1}^{n} S_i$ is a single path from the root to some leaf of T. In other words, if each player P_i plays according to one of her strategies $S_i \in \Sigma_i$, then the game follows the path determined by $\cap_{i=1}^{n} S_i$. The payoff vector that labels the terminal vertex of this path is $\vec{p}(S_1, S_2, \ldots, S_n) = (p_i(S_1, S_2, \ldots, S_n))$.

Looking again at our example, it is easy to verify that if Bonnie plays according to her strategy $S_{B,c}$ and Clyde plays according to his strategy $S_{C,q}$, then the game follows the path $root \rightarrow 1 \rightarrow 1.2$ and results in the payoff vector $\vec{p}(S_{B,c}, S_{C,q}) = (-1, -10)$.

2.2 Equilibria

The goal of the players in a game is to maximize their payoffs. A good strategy drives its user toward this goal. Indeed, a player P in a game may have a strategy S^* that is the *best response* to any combination of strategies played by the other players, in the sense that P's payoff is maximal if she plays S^* – no matter what strategies are played by the others. This strategy is called *dominant strategy for P* and formalized as follows:

Definition 2.3 *Let T be a game tree with players $P_1, P_2, \ldots, P_n$ and strategy sets $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$ for the players. $S_i^* \in \Sigma_i$ is a* dominant strategy for P_i *if for all $S_1 \in \Sigma_1, S_2 \in \Sigma_2, \ldots, S_n \in \Sigma_n$ (where $S_i \neq S_i^*$)*

$$p_i(S_1, S_2, \ldots, S_i, \ldots, S_n) < p_i(S_1, S_2, \ldots, S_i^*, \ldots, S_n)$$

If each of the n players has a dominant strategy, then the n-tuple of these is called *dominant strategy equilibrium.*

Definition 2.4 *Let T be a game tree with players $P_1, P_2, \ldots, P_n$ and strategy sets $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$ for the players. An n-tuple of strategies $(S_1^*, S_2^*, \ldots, S_n^*) \in \Sigma_1 \times \Sigma_2 \times \ldots \times \Sigma_n$ is a* dominant strategy equilibrium *if S_i^* is a dominant strategy for P_i for all $1 \leq i \leq n$.*

The pair of strategies $(S_{B,c}, S_{C,c})$ in the game of The Prisoner's Dilemma is a dominant strategy equilibrium, because, as we explained before, confessing is the best thing each player can do.

There are, however, games, in which no dominant strategy exists for any of the players. In these games, one has to apply other equilibrium concepts to find the n-tuple that contains the "best" strategies for the players. Another frequently encountered equilibrium concept is called *Nash equilibrium.* An n-tuple of strategies is a Nash equilibrium if no player has incentive to deviate from his strategy assuming that the other players do not deviate. The formal definition is the following:

Definition 2.5 *Let T be a game tree with players $P_1, P_2, \ldots, P_n$ and strategy sets $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$ for the players. An n-tuple of strategies $(S_1^*, S_2^*, \ldots, S_n^*) \in \Sigma_1 \times \Sigma_2 \times \ldots \times \Sigma_n$ is a* Nash equilibrium *if for all $1 \leq i \leq n$ and for all $S_i \in \Sigma_i$*

$$p_i(S_1^*, S_2^*, \ldots, S_i, \ldots, S_n^*) \leq p_i(S_1^*, S_2^*, \ldots, S_i^*, \ldots, S_n^*)$$

It is easy to verify that the pair of strategies $(S_{B,c}, S_{C,c})$ in The Prisoner's Dilemma is a Nash equilibrium and it is the only one.

There are several other equilibrium concepts, such as weak dominant strategy equilibrium and iterated-dominant strategy equilibrium, which we do not introduce here because we do not use them in this paper.

3 Modeling exchange protocols using game theory

There are some striking similarities between exchanges and games. Indeed, in both cases, we have two (or more) **parties/players** who interact with each other according to some **rules**, and whose **actions** influence the future actions of the other; in the course of the interaction, they may or may not have **perfect information**; the goal of the parties/players is to maximize their **profit/payoff**; in order to achieve this, they choose and follow a **strategy** (which, in the case of exchanges, may or may not coincide with the faithful execution of a given exchange protocol). This inspired us to use game theory as a tool to analyze the properties of exchange protocols.

The following conventions can be used when transforming an exchange protocol in a game:

- **Players.** The participants of the exchange protocol are represented by the players of the game.

- **Game tree.** When a protocol participant is about to send a message, she has several options: to send the message correctly, to send it incorrectly, or to not send it at all. Indeed, a participant may send any kind and number of messages that she can construct using the information she possesses at that time. These may be represented as possible moves in the game and determine the structure of the game tree.

- **Payoffs.** An advantage can be represented with a positive payoff, and a disadvantage can be modeled as a negative payoff. Let us assume, for instance, that the participants of the exchange protocol are called A and B, and the items that they want to exchange are i_A and i_B, respectively. Then, one way to compute the payoff for A may be the following: If A loses control[2] over i_A, but, as a compensation, gains access to i_B, or she does not lose control over i_A and she does not gain access to i_B, then the payoff for A is 0. The rationale is that, in the two cases described above, A does not lose or gain any advantages, and the payoff of 0 represents this situation. If, on the other hand, A gains access to i_B and does not lose control over i_A, then the payoff for A is 1, since, in this case, A has an advantage. Similarly, if A does not gain access to i_B, but loses control over i_A, then the payoff for A is -1, because she has a disadvantage. The payoff for B can be computed in the same way.

We hasten to note that these are only conventions and not precise rules for the transformation of any exchange protocol in a game. We by no means want to give a complete formal procedure for this transformation, mainly because we think that it would be quite difficult. Therefore, for some protocols, the determination of the players, the construction of the game tree, and the calculation of the payoffs may be different from the way we have just indicated. It may not be necessary, for instance, to explicitly model each protocol participant as a player. Usually, it is sufficient to model only those protocol participants as players that have choices and make decisions during the execution of the protocol. A trusted third party, for instance, is usually not modeled explicitly as a player, because it is assumed to execute the protocol faithfully and, thus, its actions are completely predetermined. Similarly, it may not be necessary to model each protocol message explicitly as a possible move in the game, and the payoffs may be calculated in another way, too. However, as shown in the examples provided hereafter, any particular exchange protocol can easily be transformed in a game.

[2] We use the term *lose control* in a general sense to express a disadvantageous situation for a player. Depending on the particular protocol and application in question, losing control over an item may mean that the other party has taken possession of it, that it has lost its value (e.g., a digital coin has been revoked), that it has expired, etc. Similarly, we use the term *gain access* in a general sense to express an advantageous situation for a player.

Example 1: As an example, let us consider a certified e-mail protocol, which ensures that either a proof-of-delivery and a proof-of-origin are available to the originator A and the recipient B, respectively, or neither the originator nor the recipient has any proof [DGLW96]. In order to achieve this, the protocol uses a trusted third party (TTP). When A wants to send a mail m to B, she first computes a cryptographic hash value of m, signs m, and encrypts the signed mail with the public key of the TTP. A then sends the encrypted and signed mail together with the cryptographic hash value to B. If B wants to read the mail, he has to sign the hash value and send the encrypted mail together with the signed hash value to the TTP. The TTP decrypts the mail, computes the cryptographic hash value, and verifies the signatures. If all the verifications are successful, then the TTP creates a proof-of-origin token and a proof-of-delivery token and makes the mail and the tokens available to A and B. Finally, A eventually fetches the proof-of-delivery and B eventually fetches the mail and the proof-of-origin from the TTP.

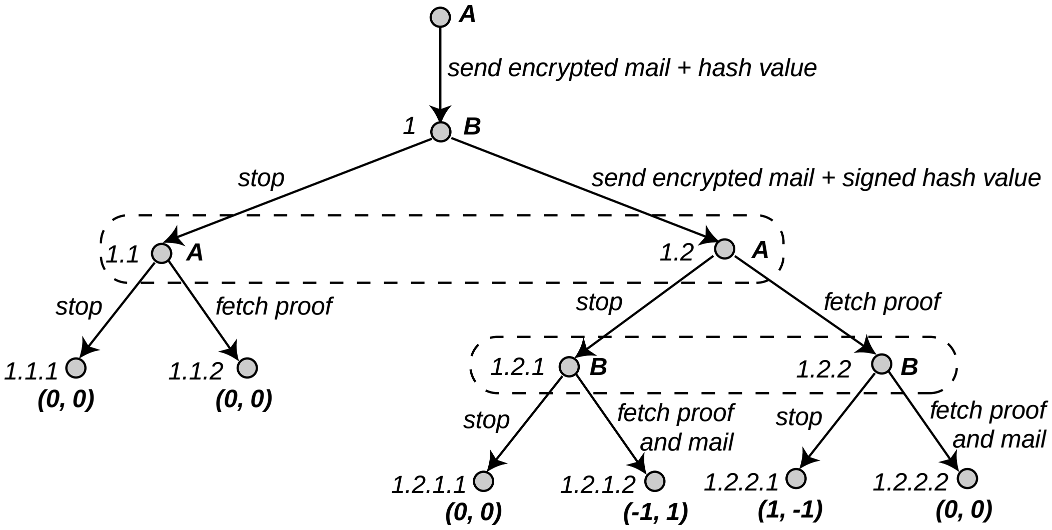

Figure 3: **Game tree for the protocol of Example 1**

The game tree for the certified mail protocol is illustrated in Figure 3. There are two players: A and B. The TTP is not a player, because it always executes the protocol faithfully, thus, it does not have choices. The game is started by A when sending the encrypted and signed mail and the hash value to B. B then has two moves: (1) either to sign the hash value and send the encrypted mail and the signed hash value correctly to the TTP, or (2) to send an incorrect message or nothing at all. These moves are represented by the two edges starting from vertex 1. Vertex 1.1 and vertex 1.2 form an information set for A, because she does not know how B moved. She has two possible moves: (1) to try to fetch the proof-of-delivery from the TTP, or (2) to stop the execution of the protocol. These are represented by the edges starting from vertex 1.1 and vertex 1.2. If B has sent an incorrect message to the TTP or stopped (i.e., if A is actually in vertex 1.1), then the TTP does not create the proof-of-delivery and the proof-of-origin tokens. This means that neither party can ever receive a proof. Therefore, independently of A's move (fetch or stop), the payoff can only be 0 for both parties. That is why both of vertex 1.1.1 and vertex 1.1.2 are labeled with the payoff vector of $(0, 0)$. If B has sent the encrypted mail and the signed hash value correctly to the TTP (i.e., if A is actually in vertex 1.2), then the TTP creates the proof tokens, and A and B may eventually get them if they try to fetch them from the TTP. Now, A and B have the same moves: each can try to fetch a proof from the TTP or stop execution. Vertex 1.2.1 and vertex 1.2.2 form an information set for B, because he does not know whether A stopped or she fetched the proof-of-delivery from the TTP. If they both fetch their proofs or they both stop, then the game terminates with a payoff of 0 for both players. These situations are represented by vertex 1.2.2.2 and vertex 1.2.1.1, respectively. If A fetches her proof, but B does not fetch his, then the payoff is 1 for A and -1 for B. Similarly, if B fetches his proof, but A does not fetch hers, then the payoff is 1 for B and -1 for A. Vertex 1.2.1.2 and vertex 1.2.2.1 represent these situations. □

One might think that every exchange protocol is best modeled with a zero-sum game (i.e., a game in which the sum of the payoffs of the players is always 0), because every situation that is advantageous for a player is disadvantageous for the other. Although this is often the case, it is not always true. Below, we present an example protocol that leads to a non-zero-sum game. In general, exchange protocols are represented by non-positive-sum games (i.e., games in which the sum of the payoffs of the players is always less than or equal to 0). We exclude those games, in which the sum of the payoffs can be positive, because they would allow the generation of wealth from nothing, which is usually not possible (as opposed to loss of wealth, which is often the case). Thus, if the game of a protocol allows a positive combined payoff, then the model is probably erroneous and should be re-considered (e.g., a party that is not modeled as a player in the game should, indeed, be modeled as a player).

Example 2: Let us consider now the following protocol that allows two parties to exchange payment for services in a way that neither party has an incentive to cheat the other [Jak95]. The protocol is based on special digital coins that can be ripped into two halves. We assume that a single half coin has no value, and once a half coin has been spent, it cannot be spent again. The protocol works as follows: First A rips a coin and sends the first half of it to B. Then B provides the service (which is worth 1 coin) to A. Finally, A sends the second half of the the coin to B, who can redeem the two halves for real money.

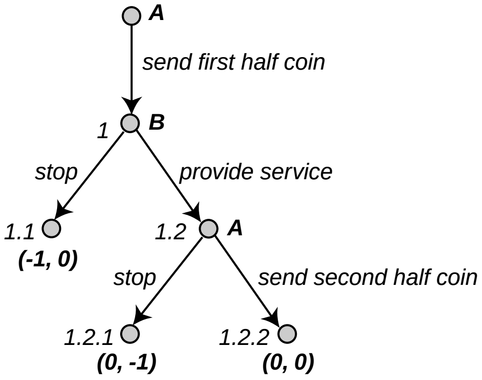

Figure 4: **Game tree for the protocol of Example 2**

The game tree of this protocol is illustrated in Figure 4. The game is started by A when sending the first half coin to B. When B receives the half coin, it has two possible moves: (1) either to provide the service, or (2) to deny service provision. These two moves are represented by the two edges starting from vertex 1. If B denies the service provision, then the game terminates in vertex 1.1 with a payoff of -1 for A and 0 for B. The reason is that, in this case, A lost her coin, although, she did not receive any services. Thus, A has a disadvantage. B neither lost nor gained anything, because he did not provide any services and he has only a half coin, which has no value. If B provides the service, then A has two choices: (1) either to send the second half coin, or (2) to stop the exchange. These choices are represented by the two edges starting from vertex 1.2. If A does not send the second half coin to B, then the game terminates in vertex 1.2.1 with a payoff of 0 for A and -1 for B. The explanation is that, in this case, A lost one coin, but she also received the service, thus, she did not lose or gain any advantages. On the other hand, B provided the service, but he has only a half coin, which is worth nothing. Thus, he has a disadvantage. If A and B execute the protocol faithfully, then the game terminates in vertex 1.2.2 with a payoff of 0 for both players for obvious reasons. □

Actually, the game model of an exchange protocol represents not only the protocol but also the possible misbehavior of the parties. Therefore, in some sense, it encodes much more than the exchange protocol itself. The exchange protocol is a set of rules that governs the interaction of the parties by specifying what they should do in any conceivable situation that may occur during the exchange. This is very similar to the concept of strategy in game theory. Indeed, we can say that the game is constructed from the description of the protocol and the assumptions about its participants and environment, and the exchange protocol itself (as a set of rules) is represented by a tuple of strategies (one strategy for each player) in the game. Therefore, fairness properties of the exchange protocol can be investigated by analyzing the properties of the strategy tuple that represents the protocol within the game. In the next section, we follow this approach to formally define various types of fairness.

4 Fairness definitions

In this section, we formalize three interpretations of the concept of fairness in two-party exchange protocols using standard notions of game theory, and investigate how these interpretations are related to each other.

4.1 Strict fairness

As we mentioned in Section 1, fairness is often defined as an atomicity property that requires a fair exchange protocol to guarantee that either both parties obtain the item of the other party, or none of them gets any useful information about the item of the other. An implicit assumption behind this definition is that at least one party behaves correctly and follows the steps of the protocol, otherwise, if none of the parties are faithful to the protocol, then the protocol can hardly guarantee anything for them. Therefore, we can say that fairness as an atomicity property means that if a party behaves correctly, then neither party can gain or lose any advantages, no matter how the other party behaves. We call this type of fairness *strict fairness* and formalize it in the following way:

Definition 4.1 *Let P be a two-party exchange protocol. Let us consider the game representation of the protocol and denote the two players by A and B and the strategy sets for the players by Σ_A and Σ_B. Let us denote the strategy pair that corresponds to the faithful execu-*

tion of the protocol by (S_A^*, S_B^*). P is said to be strictly fair *if* $\forall S_B \in \Sigma_B : \vec{p}(S_A^*, S_B) = (0,0)$ *and* $\forall S_A \in \Sigma_A : \vec{p}(S_A, S_B^*) = (0,0)$.

Protocols that try to achieve strict fairness should cope with irrationally behaving parties. An irrationally behaving party is a party that does not follow the protocol, but instead of gaining some advantages with the misbehavior, it suffers a disadvantage and may bring the other correctly behaving party in an advantageous situation. As an example, let us consider the protocol of Example 1 in Section 3. Let us assume that the originator A sends the encrypted mail and the hash value to the receiver B, who signs the hash value and sends the encrypted mail and the signed hash value to the TTP. The TTP, thus creates the proof tokens and makes them available to A and B. If at this point, A stops and does not fetch the proof-of-delivery, while B continues and fetches the mail and the proof-of-origin, then A suffers a disadvantage and B, who followed the protocol faithfully, gains an advantage. This should not happen if the protocol is strictly fair.

4.2 Safe fairness

Although theoretically possible, irrational behavior is not very frequent in practice. This means that protocols that do not try to protect the interests of irrational parties, and thus achieve weaker types of fairness (than strict fairness) can still be useful. Furthermore, since they provide weaker guarantees, they are probably less complex. This reasoning leads to the following, less demanding definition of fairness: a fair exchange protocol is a protocol that ensures that a correctly behaving party can never suffer a disadvantage. In other words, executing the protocol is safe for each party. Therefore, we call this interpretation of fairness *safe fairness* and formalize it in the following way:

Definition 4.2 *Let P be a two-party exchange protocol. Let us consider the game representation of the protocol and denote the two players by A and B and the strategy sets for the players by Σ_A and Σ_B. Let us denote the strategy pair that corresponds to the faithful execution of the protocol by (S_A^*, S_B^*). P is said to be* safe fair *if*

1. $\vec{p}(S_A^*, S_B^*) = (0,0)$, *and*

2. $\forall S_B \in \Sigma_B : p_A(S_A^*, S_B) \geq 0$ *and* $\forall S_A \in \Sigma_A : p_B(S_A, S_B^*) \geq 0$.

It is easy to verify that the certified mail protocol of Example 1 achieves this type of fairness. Note that according to the definition, a safe fair protocol may allow a misbehaving party to gain an advantage, while ensuring that a correctly behaving party does not suffer a disadvantage. However, this never happens, because correct models are always non-positive-sum games.

4.3 Nash equilibrium fairness

Instead of requiring the protocol to ensure that a correctly behaving party never suffers a disadvantage, some interpretations of fairness require the fair exchange protocol to guarantee that a misbehaving party can never gain an advantage (given that the other party behaves correctly). This idea can be formalized in a very similar way as safe fairness, but instead of requiring that $\forall S_B \in \Sigma_B : p_A(S_A^*, S_B) \geq 0$ and $\forall S_A \in \Sigma_A : p_B(S_A, S_B^*) \geq 0$, we now require that $\forall S_B \in \Sigma_B : p_B(S_A^*, S_B) \leq 0$ and $\forall S_A \in \Sigma_A : p_A(S_A, S_B^*) \leq 0$. This means that (S_A^*, S_B^*)

is a Nash equilibrium since we also require that $\vec{p}(S_A^*, S_B^*) = (0,0)$. Therefore, we call this type of fairness *Nash equilibrium fairness* and formalize it in the following way:

Definition 4.3 *Let P be a two-party exchange protocol. Let us consider the game representation of the protocol and denote the two players by A and B. Let us denote the strategy pair that corresponds to the faithful execution of the protocol by (S_A^*, S_B^*). P is said to be* Nash equilibrium fair *if*

1. $\vec{p}(S_A^*, S_B^*) = (0,0)$*, and*
2. (S_A^*, S_B^*) *is a Nash equilibrium.*

It is easy to verify that the protocol of Example 2 provides this type of fairness. As it can be seen from the definition, Nash equilibrium fair protocols do not guarantee that a correctly behaving party never suffers a disadvantage as a consequence of the misbehavior of the other party. They do, however, make misbehaving uninteresting for each party, because they ensure that the misbehaving party loses something as well, or at least it does not gain anything (apart from malicious joy) with the misbehavior.

4.4 Relationships

A direct consequence of the definitions is that strict fairness implies both safe fairness and Nash equilibrium fairness, but not vice versa. Therefore, strict fairness is stronger than safe fairness and Nash equilibrium fairness.

The relationship between safe fairness and Nash equilibrium fairness is summarized in the following two theorems:

Theorem 4.1 *Safe fairness implies Nash equilibrium fairness given that the game representation of the two-party exchange protocol is a non-positive-sum game.*

Proof: The proof follows directly from the fact that if the game is non-positive-sum, then for all $S_A \in \Sigma_A$ and for all $S_B \in \Sigma_B$, $p_A(S_A, S_B) \geq 0$ implies $p_B(S_A, S_B) \leq 0$ and $p_B(S_A, S_B) \geq 0$ implies $p_A(S_A, S_B) \leq 0$. $\Box$

Theorem 4.2 *Nash equilibrium fairness implies safe fairness if, and only if, the game representation of the two-party exchange protocol is a zero-sum game.*

Proof: First, we prove that Nash equilibrium fairness implies safe fairness if the game representation of the protocol is zero-sum. This follows directly from the fact that if the game is zero-sum, then for all $S_A \in \Sigma_A$ and for all $S_B \in \Sigma_B$, $p_A(S_A, S_B) \leq 0$ implies $p_B(S_A, S_B) \geq 0$ and $p_B(S_A, S_B) \leq 0$ implies $p_A(S_A, S_B) \geq 0$. Next, we have to prove that if the game is not zero-sum, then Nash equilibrium fairness does not imply safe fairness. This follows from the fact that there are protocols that are Nash equilibrium fair but not safe fair; an example is the protocol of Example 2 in Section 3. $\Box$

Thus, in general, safe fairness is stronger than Nash equilibrium fairness, but they coincide if the game model of the protocol is zero-sum.

5 Related work

In spite of the crucial importance of fair exchange, very few researchers have tried to produce a formal definition of this concept. We report here on two attempts in this direction, and compare them with our proposal.

In [Aso98], Asokan defines two types of fairness:

- *strong fairness*: When the protocol has completed, either each party has the item of the other party, or neither party has gained any additional information about the item of the other.
- *weak fairness*: When the protocol has completed, either strong fairness is achieved, or a correctly behaving party, say P, can prove to an arbiter that the other party has received (or can still receive) P's item without any further intervention from P.

Both strong and weak fairness consider fairness as some sort of atomicity property. They differ in the items actually exchanged in the protocol: strong fairness guarantees that the real items are exchanged, whereas weak fairness only ensures that either the real items are exchanged or one party receives the real item and the other party receives an affidavit, which can be used later and outside the system to eventually obtain the real item (typically by the enforcement from an authority). Thus, strong and weak fairness are concerned with the question: *how* can fairness (as an atomicity property) be achieved? In case of strong fairness, the atomic property of the exchange is guaranteed by the protocol itself. In case of weak fairness, it may be achieved only with help coming from outside of the system. As opposed to the approach of Asokan, in this paper, we are mainly concerned with the question of *what* fairness is, and less interested in the way it can be achieved. This means that our work complements rather than extends or refines the work of Asokan. In fact, as they cover complementary aspects, both proposals can be combined, and we carefully selected the terminology of the definitions provided in Section 4 in order to prevent ambiguities.

Formal definitions for strong fairness and weak fairness are given by Gärtner *et al.* in [GPV99, PG99]. They adopt the formalism of concurrency theory and define strong fairness and weak fairness based on safety and liveness properties. Strong fairness is defined as a safety property [PG99]. Weak fairness can be defined as a liveness or as a safety property depending on whether the dispute is resolved within the system or outside the system [GPV99]. If dispute resolution is possible within the system, then weak fairness can be formalised as a liveness property. The authors call this type of weak fairness *eventually strong fairness*. If dispute resolution is not possible within the system, then weak fairness is formalised as a safety property. Although their proposal has certainly a strong potential, it is somewhat limited to the study of strong and weak fairness as they were defined by Asokan. The advantages of their approach are the very precise system model and the possibility to derive requirements on the system that are necessary to achieve strong fairness. The latter is important when considering implementation issues.

6 Conclusion and extensions

In this paper, we introduced the idea of using game theory as a formal framework in which different types of exchange protocols can be modeled and formal definitions for fairness can be given. We illustrated the use of game theory in this context in two ways:

1. We modeled exchange protocols with game trees.
2. We formally defined various types of fairness using standard notions of game theory, and showed how the defined fairness types are related to each other.

The strongest form of fairness we introduced is called *strict fairness*. An exchange protocol that achieves this type of fairness ensures that neither party can gain or lose any advantages assuming that at least one of the parties behaves correctly. A weaker type of fairness is called *safe fairness*. An exchange protocol is said to be safe fair if it guarantees that a correctly behaving party can never suffer a disadvantage. Finally, the weakest type of fairness we introduced is called *Nash equilibrium fairness*. A protocol achieves this if it ensures that a misbehaving party cannot gain an advantage by the misbehavior assuming that the other party behaves correctly.

We believe that each of these definitions are meaningful and capture a valid interpretation of the concept of fairness in electronic commerce. Their existence is a warning that fairness may be understood in several different ways and it is important, when specifying a fair exchange protocol, to make it clear which type of fairness is provided. Furthermore, the formal definitions of different types of fairness make it possible to understand their nature better and compare them easier. Indeed, we can now establish a classification of existing and future fair exchange protocols and, therefore, assess their strengths and weaknesses. Since, once the game tree representation of a fair exchange protocol is given, the verification whether the protocol satisfies the conditions of the various fairness definitions is rather mechanical, these formal definitions may also serve as the basis of an automated analysis tool for fair exchange protocols.

Beside clarifying the concept of fairness, the formal definitions and, in particular, the application of game theory may have additional advantages: they can suggest as yet unknown (or not studied) types of fairness. One can easily define, for instance, *dominant strategy equilibrium fairness* by slightly changing the definition of Nash equilibrium fairness and requiring that the strategy pair that corresponds to the faithful execution of the protocol be a dominant strategy equilibrium. This yields a stronger type of fairness than Nash equilibrium fairness, and interestingly, in the case of protocols that can be modeled as zero-sum games, it is also stronger than safe fairness (i.e., it implies safe fairness, but not vice versa).

Furthermore, one can also define probabilistic versions of our fairness types. Probability can be introduced into the model in two ways. First, we can introduce a pseudo-player, which we may call *Chance* or *Nature*, who would choose its moves according to some probability distribution. Second, we can allow participants to play *mixed strategies*, which means that each participant picks a strategy according to some probability distribution. In both cases, we can assign a probability to each leaf of the game tree that models the protocol and give fairness definitions in terms of expected payoffs. This would make sense in case of repeated exchanges between the same two parties (e.g., in case of micropayments).

There are other ways as well to extend our results. One obvious direction would be to generalize our fairness definitions for n-party protocols. Another less obvious extension is the following: In our definitions, we implicitly assumed that there is always one single strategy for each player that corresponds to the faithful execution of the protocol. In general however, this is not necessarily true, because the protocol itself may offer several legitimate choices to a participant, which may result in several faithful strategies for that participant. In spite of the fact that we are not aware of any fair exchange protocols of this type, in principle, such protocols may exist. Our fairness definitions can be generalized for these protocols.

References

[Aso98] N. Asokan. *Fairness in Electronic Commerce.* PhD thesis, University of Waterloo, Ontario, Canada, May 1998.

[ASW97] N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, April 1997.

[BDM98] F. Bao, R. Deng, and W. Mao. Efficient and practical fair exchange protocols with off-line TTP. In *Proceedings of the 1998 IEEE Symposium on Security and Privacy*, 1998.

[But99] L. Buttyán. Removing the financial incentive to cheat in micropayment schemes. *IEE Electronics Letters*, 36(2):132–133, January 2000.

[Cle89] R. Cleve. Controlled gradual disclosure schemes for random bits and their applications. In *Proceedings of CRYPTO'89*, pages 573–588, 1990.

[DGLW96] R. Deng, L. Gong, A. Lazar, and W. Wang. Practical protocols for certified electronic mail. *Journal of Network and Systems Management*, 4(3), pages 279–297, 1996.

[FR97] M. Franklin and M. Reiter. Fair exchange with a semi-trusted third party. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 1–6, April 1997.

[GPV99] F. Gaertner, H. Pagnia, and H. Vogt. Approaching a formal definition of fairness in electronic commerce. In *Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems (Workshop on Electronic Commerce)*, pages 354–359, October 1999.

[Jak95] M. Jakobsson. Ripping coins for a fair exchange. In *Proceedings of EUROCRYPT'95*, pages 220–230, 1995.

[Ket95] S. Ketchpel. Transaction protection for information buyers and sellers. In *Proceedings of DAGS'95, Electronic Publishing and the Information Superhighway*, June 1995.

[Mor94] P. Morris. *Introduction to Game Theory.* Springer-Verlag, 1994.

[PG99] H. Pagnia and F. Gaertner. On the impossibility of fair exchange without a trusted third party. Technical Report TUD-BS-1999-02, Darmstadt University of Technology, Department of Computer Science, March 1999.

[PJ97] H. Pagnia and R. Jansen. Towards multiple-payment schemes for digital money. In *Proceedings of the First International Conference on Financial Cryptography (FC'97)*, February 1997.

[Syv98] P. Syverson. Weakly secret bit commitment: applications to lotteries and fair exchange. In *Proceedings of the 11th IEEE Computer Security Foundations Workshop*, pages 2–13, June 1998.

[Tyg96] D. Tygar. Atomicity in electronic commerce. In *Proceedings of the 15th ACM Symposium on Principles of Distributed Computing*, pages 8–26, ACM Press, May 1996.

[ZG96] J. Zhou and D. Gollmann. A fair non-repudiation protocol. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 55–61, May 1996.