# Optimistic Fair Exchange with Multiple Arbiters

Alptekin Küpçü and Anna Lysyanskaya
Brown University, Providence, RI, USA
{kupcu,anna}@cs.brown.edu

**Abstract**

Fair exchange is one of the most fundamental problems in secure distributed computation. Alice has something that Bob wants, and Bob has something that Alice wants. A fair exchange protocol would guarantee that, even if one of them maliciously deviates from the protocol, either both of them get the desired content, or neither of them do. It is known that no two-party protocol can guarantee fairness in general; therefore the presence of a trusted *arbiter* is necessary. In optimistic fair exchange, the arbiter only gets involved in case of faults, but needs to be trusted. To reduce the trust put in the arbiter, it is natural to consider employing multiple arbiters.

Expensive techniques like byzantine agreement or secure multi-party computation with $\Omega(n^2)$ communication can be applied to distribute arbiters in a non-autonomous way. Yet we are interested in efficient protocols that can be achieved by keeping the arbiters autonomous (non-communicating), especially for p2p settings in which the arbiters do not even know each other. Avoine and Vaudenay [6] employ multiple autonomous arbiters in their optimistic fair exchange protocol which uses global timeout mechanisms; all arbiters have access to -loosely- synchronized clocks. They left two open questions regarding the use of distributed autonomous arbiters: (1) Can an optimistic fair exchange protocol without timeouts provide fairness (since it is hard to achieve synchronization in a p2p setting) when employing multiple autonomous arbiters? (2) Can any other optimistic fair exchange protocol with timeouts achieve better bounds on the number of honest arbiters required? In this paper, we answer both questions negatively. To answer these questions, we define a general class of optimistic fair exchange protocols with multiple arbiters, called "distributed arbiter fair exchange" (DAFE) protocols. Informally, in a DAFE protocol, if a participant fails to send a correctly formed message, the other party must contact some subset of the arbiters and get correctly formed responses from them. The arbiters do not communicate with each other, but only to Alice and Bob. We prove that no DAFE protocol can meaningfully exist.

**Keywords**: optimistic fair exchange, distributed arbiters, trusted third party.

## 1 Introduction

Optimistic fair exchange is a very useful primitive in distributed system design with many applications including contract signing, electronic commerce, or even peer-to-peer file sharing [2, 3, 4, 5, 7, 8, 15, 18, 19, 20]. In a fair exchange protocol, Alice and Bob want to exchange some items, and they want to do so fairly. Fairness intuitively refers to Alice getting Bob's item and Bob getting Alice's item at the end of the protocol, or neither of them getting anything, even if one of them maliciously deviates from the protocol. For technical definitions of optimistic fair exchange protocols, we refer the reader to [18].

It has been shown that no general fair exchange protocol can provide complete fairness without a trusted entity [21], called the *arbiter*. In an optimistic fair exchange protocol, the arbiter is not involved unless there is a dispute between the participants. But having a single trusted entity is one of the biggest problems that make the use of such protocols hard in practice. Therefore, the use of multiple arbiters is generally motivated by reducing the trust put on the arbiter [6, 18].[1] A very natural question is how to achieve fairness in the absence of a single trusted arbiter; for example, what if we have $n$ arbiters only a fraction of whom we want to put our trust in? It is clear that this can be achieved using byzantine agreement or secure multi-party computation techniques [17, 9, 10, 14] with $\Omega(n^2)$ communication, but can we do better than that? In particular, can we do anything in a setting where the arbiters need not communicate with each other to resolve disputes? This issue is highly relevant especially for peer-to-peer settings in which the arbiters do not even know each other, and may not have enough resources for complicated schemes. Furthermore, if the scheme gets more costly, it will be hard to incentivize multiple arbiters to arbitrate, since they will get overloaded.

Avoine and Vaudenay (AV) [6] address this problem in their paper by using verifiable secret sharing techniques to employ multiple arbiters in their fair exchange protocol for a p2p system. In their setting, two peers are performing a fair exchange, and a number of other peers constitute the arbiters. They provide bounds on the number of arbiters that should be honest for their protocol to be fair (see Section 7). A crucial point is that the protocol uses global timeout mechanisms, which assumes all arbiters have access to -loosely- synchronized clocks, and the arbiters are autonomous (they do not communicate with each other). They leave two important issues as open questions: (1) Can an optimistic fair exchange protocol without timeouts provide fairness (since it is hard to achieve synchronization in a p2p setting) when employing multiple autonomous arbiters? (2) Can any other optimistic fair exchange protocol with timeouts achieve better bounds on the number of arbiters that need to be honest?

Unfortunately, in this paper, we answer both of these questions negatively. Inspired by state-of-the-art optimistic fair exchange protocols with a single arbiter, we define a general class of optimistic fair exchange protocols with multiple arbiters, called "distributed arbiter fair exchange" (DAFE) protocols. Informally, in a DAFE protocol, if one of the participants fails to send a correctly formed message, the other participant must contact some subset of the arbiters and get correctly formed responses from them in order to make the exchange fair.[2] Two main properties of a DAFE protocol are its abort/resolve semantics and the autonomy of multiple arbiters used, as discussed in Section 2. In a DAFE protocol, the arbiters are autonomous; they do not talk to each other, but talk only to Alice and Bob. A third property is the state machine semantics of the participants. We show that this class of protocols capture currently known state-of-the-art optimistic fair exchange protocols extended to use multiple distributed arbiters in a very intuitive manner, as shown in Section 2.1. Under this framework, in Section 4 we analyze scenarios that can occur during the execution of instances of optimistic fair exchange protocols, and prove some predicates every such protocol must satisfy to be able to provide semantic fairness, which is a property that needs to be satisfied by all optimistic fair exchange protocols.

In Section 5, we prove that no DAFE protocol can provide fairness meaningfully[3], answering

[1] It is possible to have multiple arbiters deployed for reducing the load, but if only one of them is employed per exchange, we do not consider that protocol as having distributed arbiters.

[2] Of course, if no message is sent yet, there is no need to contact arbiters, which is not an interesting case to analyze anyway.

[3] We prove that multiple arbiters are no better (or actually worse) than a single arbiter in terms of trust in the

the first open question negatively. In Section 6, we prove impossibility of DAFE protocols using threshold-based mechanisms (any $k$ arbiters are enough for resolution) even when the autonomous arbiters assumption is relaxed. For protocols using general set-based mechanisms (any $k$ arbiters will not be enough for resolution, specific sets of arbiters need to be contacted), we cannot prove impossibility in this relaxed setting, but we conjecture that such protocols are not possible. However, our impossibility results can be overcome in the timeout model (where all arbiters have access to loosely synchronized clocks) and also in case the arbiters can communicate. We use our framework to analyze the existing AV protocol [6] in this timeout model in Section 7, showing how easy it is to apply our framework. We prove that the bounds on the required number of honest arbiters proven earlier for that protocol are optimal, and hence answer the second open question also negatively.

These results mean that many optimistic fair exchange protocols that want to efficiently distribute their arbiters may need to employ synchronized clocks. And even in this case, they cannot hope to require fewer honest arbiters than the Avoine and Vaudenay protocol [6]. If they do not want to employ synchronized clocks, then they may need to employ costly solutions like secure multi-party computation or Byzantine agreement.

## 2 Definition of a DAFE protocol

In this section, we define a general optimistic fair exchange model that fits currently known state-of-the-art optimistic fair exchange schemes that uses an arbiter, and has semantics for aborting and resolving that we define below.

All the participants (Alice, Bob and the arbiters) are interactive Turing Machines (ITMs)[4]. Those ITMs have the following 4 semantic states: *Working* , *Aborted* , *Resolved* , *Dispute* (see Figure 1). These semantic states can correspond to multiple states in the actual ITM definitions of the participants, but these abstractions will be used to prove our results.

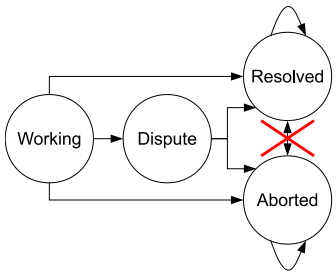


Figure 1: Semantic view of the state machines of the participants.

The ITM of each participant starts in the *Working* state. Semantically, *Working* state denotes any state that the actual ITM of a participant is in when the protocol is still taking place. When a participant does not receive the expected correctly formed message from the other participant, he can possibly abort or decide to contact the arbiters for resolving or aborting with them, in which case the ITM of that participant enters its *Dispute* state. If everything goes well in the protocol execution (all messages received from the other party are correctly formed), then the ITM of a participant transitions to the *Resolved* state directly from the *Working* state. Otherwise, if the arbiters needed to be contacted, the ITM first visits the *Dispute* state, and then transitions to either *Resolved* or *Aborted* state. Arbiters' *Dispute* state is dummy, and hence not needed in our analysis. Furthermore, when in Section 6 we relax one of our assumptions, even Alice and Bob will not have this *Dispute* state.

When the protocol ends, Alice and Bob are allowed to end only in *Aborted* or *Resolved* states. If Alice or Bob ends at its *Resolved* state, then, by definition, (s)he must have obtained the exchange item from the other party. When the protocol ends, if the ITM of a participant is not in its *Resolved* state, it is considered to be in its *Aborted* state.

DAFE framework.

[4]The ITMs have access to –possibly synchronized– clocks for timeout mechanisms.

Using these semantic definitions, even an adversarial ITM can be considered to have those 4 states (since it either obtains the other party's item and hence ends at its *Resolved* state, or not therefore ending at its *Aborted* state). The adversarial ITM does not necessarily have a *Dispute* state, but this will not affect any results presented in this paper. One can think that the moment the honest party's ITM enters its *Dispute* state, the adversarial ITM also enters its *Dispute* state.

We will talk about only complete DAFE protocols (for a definition of optimistic fair exchange protocols, see Appendix A): when both participants are honest, they end at their *Resolved* states. Since our goal here is to analyze fairness of such protocols, the only interesting case is when we have one honest party denoted $H$ and one malicious party denoted $M$. We will not consider cases where both parties are malicious since there is no honest party to protect.

**Definition 1** (End of the Protocol)**.** *We say that the protocol has ended if (1) the honest party ended up being in her either* Resolved *or* Aborted *state, and (2) the adversary produced its final output at its either* Resolved *or* Aborted *state after running at most a polynomial number of steps (polynomial in some security parameter).*

Now that we defined our participants carefully, we can state our assumptions on them and define DAFE protocols.

Distributed Arbiter Fair Exchange (DAFE) protocols: DAFE protocols are optimistic fair exchange protocols that can be characterized with the following:

- Exclusive states assumption
- Connection between arbiters' state and Alice's and Bob's
- Autonomous arbiters assumption

Exclusive states assumption: This assumption states that the *Resolved* and *Aborted* states are mutually exclusive. For an arbiter, those states informally mean whether or not the arbiter helped one of the parties to resolve or abort. We assume that there is no combination of state transitions that can take an honest *arbiter* from the *Aborted* state to the *Resolved* state, or vice versa. In most existing protocols, this corresponds to the fact that the arbiter will not abort with a participant first and then decide to resolve with him or the other participant, or vice versa. An honest arbiter can keep executing abort (or resolve) protocols with other participants in the exchange while he is in the *Aborted* (or *Resolved* , respectively) state, but can not switch between states for different participants.

**Definition 2** (Aborting and Resolving with an Arbiter)**.** *If a participant interacts with an arbiter and aborts with him, the arbiter goes to his* Aborted *state, from where he will never switch to his* Resolved *state. Similarly, if a participant resolves with an arbiter, the arbiter goes to his* Resolved *state, from where he will never switch to his* Aborted *state.*[5]

**Definition 3** (Aborted and Resolved Protocol Instance)**.** *A protocol instance is called aborted if both Alice and Bob ended at their* Aborted *states, and called resolved if both Alice and Bob ended at their* Resolved *states.*

[5]Due to the exclusive states assumption, these happen only if an arbiter is not already in his *Resolved* or *Aborted* state, respectively.

Connection between arbiters' state and Alice's and Bob's: A resolution makes sense if at least one of the parties has not resolved yet. In such a case, Alice or Bob can end in their *Resolved* states (unless they already are in their *Aborted* states) only if a set of arbiters end in their *Resolved* states. This set of arbiters can be different for Alice or Bob. Actually, there can be more than one set of arbiters that is enough for this resolution. All these will be clear in later sections when we define those sets of arbiters that will be sufficient for resolution.

Autonomous arbiters assumption: We assume that the honest arbiters' decisions are made autonomously, without taking into account the decisions of the other arbiters. Arbiters can arrive at the same decision seeing the same input, but they will not consider each other's decision while making their own decisions. In particular, this means no communication takes place between honest arbiters (malicious arbiters can do anything they want).

Our goal in this is to distribute the trust efficiently. Without autonomy, byzantine fault tolerance or secure multiparty computation techniques [17, 9, 10, 14] can be applied, yielding costly solutions ($\Omega(n^2)$ communication when $n$ arbiters are employed). Furthermore, autonomy of the arbiters render the deployment of such a real system practical, since no coordination of the arbiters is necessary.

Yet, a dependence between the arbiters' decisions can be generated by Alice or Bob, by contacting the arbiters with some specific order. Therefore, to model the autonomy, we require the protocol design to direct the honest participants to contact all the arbiters without any order. More formally, when the ITM of an honest participant decides to contact the arbiters for dispute resolution, the participant creates the message to send to all of the arbiters before receiving any response from any arbiter. One can model this with the *Dispute* state in which the message to send to the arbiters are prepared all at once. We will call this simultaneous (or unordered) resolve/abort. Note that this only constrains honest Alice or Bob. A malicious party can introduce dependence between messages to arbiters and responses from other arbiters. Later in Section 6 we will relax this autonomy assumption and discuss its consequences. We realize that this assumption is not necessary for most of our results, but helps making the presentation clearer.

All optimistic fair exchange protocols need to satisfy the following semantic fairness property.

Semantic Fairness: The semantic fairness property states that at the end of the protocol, Alice and Bob both end at the same state (they both end at their *Aborted* states, or they both end at their *Resolved* states). In other words, we need the protocol instance to be either resolved or aborted as in Definition 3, for every possible instance of the protocol.[6]

Optimistic fair exchange protocols should also satisfy the *timely resolution* property, meaning that the honest party need not wait indefinitely for any message from any other party. He can have a local timeout mechanism with which he can decide to proceed without waiting. In particular, he can end his side of the protocol any time he wants, ending at his *Resolved* or *Aborted* state, according to the rules we defined above. Note that in general providing timely resolution guarantees necessitates mutually exclusive *Resolved* and *Aborted* states, and a way for the arbiters to transition to their *Aborted* states through interaction with other parties or through the use of timeouts.

Regular DAFE protocols do not have global timeout mechanisms, and the sets of arbiters that Alice or Bob can resolve with are well-defined by the protocol, and does not change once the honest party is in its *Dispute* state. We will show an extended version called DAFE with timeouts

[6]There will not be any cases where the honest party ends at its *Resolved* state whereas the malicious party ends at its *Aborted* state and this affects our results. Therefore, this semantic fairness definition is enough for our purposes. Furthermore, it is subjective whether or not to consider a case where two parties end at different states as fair.

(DAFET) where the protocols are allowed to use timeouts. At the timeout specified by the protocol, honest arbiters transition into their *Aborted* states. This is done using the (loosely synchronized) clocks of the ITMs. We call this event "an arbiter timeouts". We allow the possible sets of arbiters to resolve with to change at this timeout. This timeout model bypasses the impossibility results for DAFE protocols. These will be clear later.

We will first provide examples of existing optimistic fair exchange protocols with intuitive extensions to employ multiple autonomous arbiters and show how they fit our DAFE classification. Then, after defining some notation, we will analyze different possible protocol instances under different scenarios, and possible protocol types. We then show that it is impossible for some common types of DAFE protocols to provide semantic fairness, thus warning researches not to pursue that direction. We also analyze some positive results using global timeout mechanisms, and prove the optimality of the bounds of the AV protocol, showing the usability of our framework for easy analysis. We then discuss the role of autonomous arbiters and timeouts in our results and elaborate on different ideas.

### 2.1 Sample DAFE Protocols

Many currently known optimistic fair exchange protocols can be considered as special cases of DAFE protocols in which there is only one arbiter. In this section, we also discuss a way to extend them to employ multiple autonomous arbiters. Unfortunately, this means, those extended protocols cannot provide fairness, as we will prove later in this paper that no DAFE protocol can provide fairness. Precisely, our impossibility result states that all arbiters need to be trusted in a DAFE protocol, hence they are not realistic. For the special single-arbiter case, this points out to the trust assumption on the arbiter.

To the best of our knowledge, all currently known optimistic fair exchange protocols adhere with our framework. As a representative of optimistic fair exchange protocols, we will analyze a protocol due to Asokan, Shoup and Waidner (ASW) [4]. They have two versions of their protocol: one version that uses timeout-based aborts (can be converted to a DAFET protocol, see Section 7), and one that does not employ timeouts (we will discuss now). It is considered one of the state-of-the-art signature exchange protocols, and is the first completely fair optimistic exchange protocol. A state-of-the-art optimistic fair exchange protocol for exchanging files are given in [18], and all our discussion here applies to that protocol too. The ASW protocol without timeouts is described below for reference:

1. Alice sends Bob a non-verifiable escrow of her signature, with a label defining how Bob's signature should look like. Bob checks if the definition is the correct definition.

2. Bob sends Alice a *verifiable* escrow of his signature, with the label defining how Alice's signature should look like and also attaching the escrow he obtained in step 1. Alice verifies the verifiable escrow. She furthermore checks if the label is formed correctly. If anything goes wrong at this step or a *message timeout* occurs, she aborts the protocols and runs AliceAbort with the Arbiter.

3. Alice sends Bob her signature. Bob verifies this signature, and stops and runs BobResolve if it does not verify or a *message timeout* occurs.

4. Bob sends Alice his signature. If the signature does not verify, Alice runs AliceResolve.

AliceAbort tells the Arbiter to consider that trade as aborted and not to honor any further resolution request on that particular trade. BobResolve gets Alice's signature by providing Bob's signature, and similarly, AliceResolve gets Bob's signature by providing Alice's signature.

In terms of the state semantics of the participants, it is clear that the ending states of the participants can be parsed into *Aborted* and *Resolved* states which are mutually exclusive. Furthermore, honest participants are not allowed to transition between *Aborted* and *Resolved* states. In particular, once Alice aborts with the arbiter taking him to his *Aborted* state, he will refuse resolving with Bob. Since there is only one arbiter, it is autonomous. As for the connection between arbiter's state and Alice's and Bob's, it is clear that in case of a dispute, their state depends on the arbiter's.

Now, if we want to extend those protocols to use multiple autonomous arbiters, one easy way is to employ verifiable secret sharing techniques [6, 22, 16]. The state-of-the-art optimistic fair exchange protocols employ verifiable escrows [11, 13, 4, 18] under the (one and only) arbiter's public key. The intuition behind using verifiable escrows is that the recipient can verify, without learning the actual content, that the encrypted content is the content that is promised and the arbiter can decrypt it. Verifiable secret sharing techniques can be employed to split the promised secret per arbiter. Each of these secrets will be encrypted under a different arbiter's public key. The recipient can still verify those encrypted shares can be decrypted and combined to obtain the promised secret, thereby effectively achieving the same goal as a verifiable escrow, but for multiple arbiters. For a detailed explanation of how to use verifiable secret sharing in distributing the arbiters, we refer the reader to [6].

When we extend the ASW protocol to use multiple autonomous arbiters, instead of this verifiable escrow, the participants will use verifiable secret sharing techniques as explained above and in [6]. Regardless of whether threshold- or set-based secret sharing mechanisms are used, the resolution procedure now requires contacting multiple arbiters. For example, if the threshold for the secret sharing method used is $k$, the resolution will involve contacting at least $k$ arbiters.

In terms of the state semantics of the participants, it is clear that the ending states of the participants can be parsed into *Aborted* and *Resolved* states which are mutually exclusive. Because we assume the arbiters are contacted simultaneously, the autonomy of the arbiters hold. As for the connection between arbiters' state and Alice's and Bob's, since resolution needs $k$ shares, and secure secret sharing and encryption methods are used, a participant can obtain the other participant's exchange item if and only if (s)he resolves with at least $k$ arbiters (in case of a dispute). This relationship makes perfect sense when multiple autonomous arbiters are used, since the main goal in distributing the arbiter is distributing the trust. Therefore, the goal is to find some number of honest arbiters each one of which will individually contribute to dispute resolution between participants by resolving or aborting with them. When arbitrary sets are used instead of thresholds, it is easy to see all these arguments will still apply.

The same techniques can be applied to another state-of-the-art optimistic fair exchange protocol [18] designed to exchange multiple files between participants. This protocol employs a verifiable escrow for escrowing the payment (endorsement of an unendorsed e-coin [12]) sent by the participants. All the arguments for the ASW protocol also apply here. Again, verifiable secret sharing techniques as discussed above will be used instead of the verifiable escrow. The resolution mechanism will be similar to the ones we described for the extended ASW protocol. As for the state semantics, a participant goes to her/his *Resolved* state if (s)he gets other participant's file or e-coin, and goes to his/her *Aborted* state otherwise.

In Section 4 we will analyze possible scenarios in an optimistic fair exchange protocol. The first two scenarios will be applicable to this extended protocol types, as we show in Section 5, where we analyze protocols that have the same structure as ASW protocol.

# 3 Notation

Remember that in a fair exchange scenario, Alice and Bob want to exchange some items fairly. In case of a dispute, they need to contact the arbiters. They are allowed to take the following two actions with the arbiters: *abort* or *resolve*. As noted in Definition 2, aborting with an honest arbiter takes him to his *Aborted* state, whereas resolving with him would take him to his *Resolved* state.[7] Remember, those states are mutually exclusive, and there is no transition between them, direct or indirect. We assume that the arbiters are autonomous: They do not take into account other arbiters' decision while acting. More formally, the honest participant contacts all arbiters simultaneously (her messages to arbiters do not depend on any response from any of the arbiters).

Let $N$ denote the set of all arbiters, where there are a total of $n$ of them ($|N| = n$). An honest arbiter acts as specified by the protocol. Let $F$ be the set of arbiters who are friends with a malicious participant. Those arbiters are adversarial.[8]

Define two sets $\mathcal{H}_\mathcal{R}$ and $\mathcal{M}_\mathcal{R}$, which are sets of sets. Any set $H_R \in \mathcal{H}_\mathcal{R}$ is a set of arbiters that is sufficient for the *honest* party to *resolve* (as defined in Section 2 during the discussion about the connection between arbiters' state and Alice's and Bob's). Similarly, any set $M_R \in \mathcal{M}_\mathcal{R}$ is a set of arbiters that is sufficient for the *malicious* party to *resolve*. Therefore, by definition, in case of a dispute, the honest party will end at her *Resolved* state **if and only if** she resolves with **all** the arbiters in **any one** of the sets in $\mathcal{H}_\mathcal{R}$ (unless she already is in her *Resolved* state). Similarly, the malicious party will end at his *Resolved* state **if and only if** he resolves with **all** the arbiters in **any one** of the sets in $\mathcal{M}_\mathcal{R}$ (unless he already is in his *Resolved* state). *For DAFE protocols, these sets are well-defined by the protocol description, and do not change once the honest party enters its* Dispute *state.*

A special case of these sets can be represented as thresholds. Let $T_H$ be the number of arbiters the honest party needs to contact for resolution. Similarly, $T_M$ denotes the number of arbiters the malicious party needs to contact for resolution. Thus, the set $\mathcal{H}_\mathcal{R}$ is composed of all subsets of $N$ with $T_H$ or more arbiters. Similarly, the set $\mathcal{M}_\mathcal{R}$ is composed of all subsets of $N$ with $T_M$ or more arbiters.

Define $R_H$ as the set of arbiters the honest party $H$ has already resolved with, and $R_M$ as the set of arbiters the malicious party $M$ has already resolved with. Also define $R_A$ as the set of all arbiters that are available for $H$ for resolution. Initially, when the dispute resolution begins, we assume that $R_H = \varnothing$, $R_M = F$, and $R_A = N - F$ (and all arbiters are available for resolution to the malicious party). We furthermore have the following actions and their effects on these sets:

**Action 1** ($H$ resolves with an arbiter $X$)**.** *The effect is that* $R_H$ *becomes* $R_H \cup \{X\}$.

**Action 2** ($M$ resolves with an arbiter $X$)**.** *The effect is that* $R_M$ *becomes* $R_M \cup \{X\}$.

**Action 3** ($H$ aborts with an arbiter $X \in R_A$)**.** *The effect is that* $R_A$ *becomes* $R_A - \{X\}$.

**Action 4** ($M$ aborts with an arbiter $X \in R_A$)**.** *The effect is that* $R_A$ *becomes* $R_A - \{X\}$.

[7] This happens only if an arbiter is not already in its *Resolved* or *Aborted* state, respectively.

[8] For example, they may appear as aborted to the honest party, but they may still resolve with the malicious party.

Note that we do not care what these sets actually are, or whether or not one can find such sets of sets. For our impossibility result, it is enough that conceptually these sets of sets exist.

As in previous work on optimistic fair exchange [4, 18], we assume that the adversary can reorder messages, delay the honest party's messages to the arbiters, insert his own messages, *etc.* But he cannot delay honest party's messages indefinitely: the honest party eventually reaches the arbiters that he wants to contact initially, and this occurs before the timeout if the protocol uses timeout mechanisms.

### 3.1 DAFET protocols (DAFE protocols with timeouts):

In DAFET protocols, we allow for timeouts by giving the arbiters access to loosely synchronized clocks. Instead of actions 3 and 4 above (honest or malicious party aborting), the following action is allowed:

**Action 5** (An arbiter $X \in R_A - R_H - R_M$ timeouts)**.** *The effect is that* $R_A$ *becomes* $R_A - \{X\}$.

Another difference between DAFE and DAFET protocols is the sets $\mathcal{H}_{\mathcal{R}}$ and $\mathcal{M}_{\mathcal{R}}$ being static and dynamic, respectively. DAFE protocols define such sets as static: the overall set of arbiters that needs to be contacted for resolution does not change with time once the honest party enters its *Dispute* state (hence the notation $\mathcal{H}_{\mathcal{R}}$ and $\mathcal{M}_{\mathcal{R}}$). In contrast, we allow DAFET protocols to employ dynamic sets (hence the notation $\mathcal{H}_{\mathcal{R}}(t)$ and $\mathcal{M}_{\mathcal{R}}(t)$). These sets may depend on the timeout and possibly the parties' actions in that particular instance of the protocol. Consider the following two cases as illustrative examples: Some type of protocols allow, let's say, Alice to resolve only after a timeout. Some other type of protocols allow Alice to resolve only with an arbiter that Bob has already resolved with (or vice versa). In analyzing such types of protocols, we will consider $\mathcal{H}_{\mathcal{R}}(t)$ and $\mathcal{M}_{\mathcal{R}}(t)$ as dynamic, letting them change with those actions. We discuss the relation between the use of timeouts and dynamic sets in fair exchange protocols more in Section 8.

We will consider any action that results in a change in those sets as new time steps, but there is no need to treat other events as separate time steps since they do not constitute a significant part of the analysis. Therefore, one can think as if any party can contact any number of arbiters at a given time step $t$. $t = 0$ denotes the time when the dispute resolution begins (the time the honest party enters its *Dispute* state, not the time the protocol execution begins).

Lastly, the set of friends of a malicious party can also change with time, if the adversary is allowed to adaptively corrupt arbiters. In that case, we will use the notation $F(t)$.

## 4 Framework for Analysis of DAFE Protocols

In this section, we will provide our framework for analyzing DAFE (and DAFET) protocols. Our framework is composed of different scenarios that can take place during the execution of an instance of a DAFE protocol. Once we have lemmas related to those scenarios stating the necessary (not necessarily sufficient) conditions that need to be satisfied so that the given scenario satisfies the semantic fairness property, then we can analyze different protocol types in the next section. For example, the extended ASW protocol discussed in the previous section will be a protocol of type 0 (in Section 5) and will employ scenarios 0 and 0 (depending on which one of Alice and Bob is malicious). Since our results are impossibility or lower bound type of results, it is enough to analyze necessary (but maybe not sufficient) conditions. In all our scenarios (except the last one),

we assume that neither party is in the *Resolved* state yet. We consider dynamic resolution sets for our scenario analysis, since static sets are a special case of dynamic sets.

### 4.1 Scenario 1: $M$ can Abort

In this scenario, we consider a protocol instance where the malicious party has the ability to abort and resolve. The honest party can abort and resolve too, but the results still apply even if he is restricted to only resolve action. In this scenario, actions 1, 2, and 4 in Section 3 are possible. Our results in this section will remain valid regardless of action 3 being possible.

**Lemma 1.** *Every DAFE protocol instance needs to make sure that there exists a time $t$ when $\forall M_R \in \mathcal{M}_\mathcal{R}(t)$ $\exists H_R \in \mathcal{H}_\mathcal{R}(t)$ s.t. $H_R \subseteq M_R - F(t)$.*

*Proof.* Assume otherwise: At any time in the protocol instance $\exists M_R \in \mathcal{M}_\mathcal{R}(t)$ *s.t.* $\forall H_R \in \mathcal{H}_\mathcal{R}(t)$ $H_R \not\subseteq M_R - F(t)$. The malicious party can break fairness as follows: He aborts with the set of arbiters $R_A - M_R$, and resolves with the set of arbiters $M_R$. Since no $H_R$ is now a subset of the available arbiters $R_A = M_R - F(t)$, the honest party cannot resolve, while the malicious party already resolved. Thus this protocol instance is unfair (does not satisfy semantic fairness). □

**Corollary 1.** *At any given time $t$ during the protocol instance before the protocol is resolved for $H$, we need $\forall M_R \in \mathcal{M}_\mathcal{R}(t)$ $M_R \not\subseteq F(t)$ since otherwise we need $\exists H_R \in \mathcal{H}_\mathcal{R}(t)$ s.t. $H_R = \varnothing$.*

**Corollary 2.** *We need a time $t$ to exist satisfying $\exists H_R \in \mathcal{H}_\mathcal{R}(t)$ s.t. $H_R \cap F(t) = \varnothing$ since otherwise the lemma cannot be satisfied ($H$ can never resolve).*

**Corollary 3.** *Using threshold-based mechanisms, we need that there exists a time $t$ that satisfies $T_H \leq T_M - |F(t)|$.*

**Corollary 4.** *Using threshold-based mechanisms, at any given time $t$ during the protocol instance before the protocol is resolved for $H$, we need $T_M > |F(t)|$ since otherwise we need $T_H \leq 0$.*

**Corollary 5.** *Using threshold-based mechanisms, we need a time $t$ to exist satisfying $T_H \leq n - |F(t)|$ since otherwise $H$ can never resolve.*

### 4.2 Scenario 2: Only $H$ can Abort

In this scenario, we assume that the malicious party has the ability to resolve only, whereas the honest party can abort and resolve. In this scenario, actions 1 to 3 in Section 3 are possible (action 4 is not possible).

**Lemma 2.** *Every DAFE protocol instance needs to make sure that there exists a time $t$ when $\forall M_R \in \mathcal{M}_\mathcal{R}(t)$ $\exists H_R \in \mathcal{H}_\mathcal{R}(t)$ s.t. $H_R \subseteq M_R - F(t)$.*

*Proof.* Assume otherwise: At any given time $\exists M_R \in \mathcal{M}_\mathcal{R}(t)$ *s.t.* $\forall H_R \in \mathcal{H}_\mathcal{R}(t)$ $H_R \not\subseteq M_R - F(t)$. The malicious party can break fairness as follows: When $H$ wants to abort the protocol, $M$ lets abort messages to all arbiters in $R_A - M_R$ to reach their destination, but intercept the messages to $M_R - F(t)$ ($F(t)$ really does not matter since his friends will help him anyways). He then resolves with $M_R$. Even if $H$ notices this, he cannot go and resolve since there is no set $H_R \in \mathcal{H}_\mathcal{R}(t)$ that will allow him to. Therefore, this protocol instance also does not satisfy semantic fairness. □

Note that Lemma 2 is the same as Lemma 1, and therefore all the corollaries apply to this scenario too.

### 4.3 Scenario 3: $H$ can Resolve only after Timeout

In this scenario, aborts can be caused by timeouts only. The malicious party can resolve before and after the timeout, but the honest party can resolve only after the timeout. Therefore, actions 2 and 5 are possible, but not 3 and 4. Action 1 is possible only after the timeout.

**Lemma 3.** *Every DAFET protocol instance needs to make sure there exists a time $t$ when $\forall M_R \in \mathcal{M}_\mathcal{R}(t)$ $\exists H_R \in \mathcal{H}_\mathcal{R}(t)$ s.t. $H_R \subseteq M_R - F(t)$.*

*Proof.* Assume otherwise: At any given time $\exists M_R \in \mathcal{M}_\mathcal{R}(t)$ *s.t.* $\forall H_R \in \mathcal{H}_\mathcal{R}(t)$ $H_R \not\subseteq M_R - F(t)$. The malicious party can break fairness as follows: $M$ resolves with $M_R$ before the timeout. When the timeout occurs, all arbiters in $R_A - R_H - R_M$ to go to their *Aborted* states ($R_H$ being the empty set), which means now $R_A = M_R - F(t)$. But $H$ cannot resolve with the remaining available arbiters and hence this protocol instance is not semantically fair. □

Note that Lemma 3 is the same as Lemma 1, and therefore all the corollaries apply to this scenario too.

### 4.4 Scenario 4: $M$ already Resolved

All of the scenarios above assumed that both $H$ and $M$ start in their *Working* states when they are performing the resolutions. Yet, it might be perfectly possible that the resolution starts at a point in the protocol where one of the parties has already resolved (and hence is in its *Resolved* state). If $H$ has already resolved, then there is no point to further analyze, since we do not care if the protocol is fair to the malicious party. But if $M$ has already resolved, then we need the following lemma to hold:

**Lemma 4.** *Every DAFE protocol instance needs to make sure that there exists a time $t$ when $\exists H_R \in \mathcal{H}_\mathcal{R}(t)$ s.t. $H_R \cap F(t) = \varnothing$.*

*Proof.* Assume at all times $\forall H_R \in \mathcal{H}_\mathcal{R}(t)$ $H_R \cap F(t) \neq \varnothing$. The malicious party has already resolved but since all possible ways to resolve for $H$ has to go through one of the malicious party's friends, he has no hope of resolving. □

This lemma corresponds to corollary 2 and hence corollary 5 also applies here.

## 5 Impossibility Results on DAFE Protocols

The previous section analyzed possible scenarios in DAFE and DAFET protocol instances. In this section, we will analyze DAFE protocol types, using the results from different scenarios that might come up in instances of such protocols. We will conclude that no DAFE protocol can provide fairness under any realistic assumption. DAFET protocols using dynamic sets are possible indeed, and we analyze an existing DAFET protocol in Section 7.

For every protocol type, we will consider the following two cases: The case where the honest player plays the role of Alice, and the case where he plays the role of Bob. We denote the set of sets for Alice to resolve as $\mathcal{A}_\mathcal{R}(t)$; similarly $\mathcal{B}_\mathcal{R}(t)$ is for Bob to resolve. The difference in types of protocols related to these sets being static or dynamic will play a big role. For ease of analysis (and since it is enough for the impossibility results in this section) we will assume the friend list $F(t)$

of the malicious party is static (does not change with time).[9] Since this is a weaker adversary, our impossibility results will also apply when we consider stronger (adaptive) adversaries.We will use $F_A$ to denote friends of a malicious Alice, and $F_B$ to denote friends of a malicious Bob.

In the DAFE protocol types below, we will consider the sets $\mathcal{A}_\mathcal{R}(t)$ and $\mathcal{B}_\mathcal{R}(t)$ as static (therefore using the notation $\mathcal{A}_\mathcal{R}, \mathcal{B}_\mathcal{R}$), which eases the use of the lemmas. With static sets, we do not need to consider different times in the protocol instance. A lemma saying there must exist a time $t$ can be simplified by just looking at the initial sets.

### 5.1 Protocol 1: Alice and Bob can Abort and Resolve

In this type of protocols, Alice is given the ability to abort and resolve, and Bob is also given the ability to abort and resolve.

**Case 1: Honest Alice *vs.* Malicious Bob**: This case falls under Scenario 1, which means (for the static case) any DAFE protocol needs to have $\forall B_R \in \mathcal{B}_\mathcal{R}\ \exists A_R \in \mathcal{A}_\mathcal{R}\ s.t.\ A_R \subseteq B_R - F_B$.

**Case 2: Malicious Alice *vs.* Honest Bob**: This case also falls under Scenario 1, which means (again for the static case) any DAFE protocol needs to have $\forall A_R \in \mathcal{A}_\mathcal{R}\ \exists B_R \in \mathcal{B}_\mathcal{R}\ s.t.\ B_R \subseteq A_R - F_A$.

These two cases lead to the conclusion that every protocol instance needs two sets $A_R \in \mathcal{A}_\mathcal{R}$ and $B_R \in \mathcal{B}_\mathcal{R}\ s.t.\ A_R = B_R \subseteq \{\text{trusted arbiters}\}$. These arbiters must be trusted, and so there is no point in distributing the arbiters. It is even worse: If any of these arbiters are corrupted, the DAFE protocol fails to be fair. Therefore, no such realistic DAFE protocol can exist.

When considering threshold-based schemes, this corresponds to the requirement that $T_B \leq T_B - F_A - F_B$, which means no party should have any friends for such a protocol to be fair. If even one arbiter is corrupted, the protocol becomes unfair. Therefore, no such realistic DAFE protocol can exist. Since set-based mechanisms cover threshold-based ones, we will not discuss threshold-based schemes separately again unless necessary. All impossibility results proven for set-based mechanisms directly apply in the context of threshold-based ones.

### 5.2 Protocol 2: Only one party can Abort

In this type of protocols, Alice is given the ability to abort and resolve, whereas Bob is given only the ability to resolve. Analysis of protocols that are symmetric to this type of protocols (where Bob can abort and resolve, and Alice can only resolve) obviously yields to the same conclusions.

**Case 1: Honest Alice *vs.* Malicious Bob**: This case falls under Scenario 2, which requires that DAFE protocols need to make sure $\forall B_R \in \mathcal{B}_\mathcal{R}\ \exists A_R \in \mathcal{A}_\mathcal{R}\ s.t.\ A_R \subseteq B_R - F_B$.

**Case 2: Malicious Alice *vs.* Honest Bob**: This case falls under Scenario 1, which means any DAFE protocol needs to have $\forall A_R \in \mathcal{A}_\mathcal{R}\ \exists B_R \in \mathcal{B}_\mathcal{R}\ s.t.\ B_R \subseteq A_R - F_A$.

We can conclude as in the previous section (Section 5.1) that every protocol instance needs two sets $A_R \in \mathcal{A}_\mathcal{R}$ and $B_R \in \mathcal{B}_\mathcal{R}\ s.t.\ A_R = B_R \subseteq \{\text{trusted arbiters}\}$. Again, this means there is no point in distributing the arbiters in terms of trust. Remember that threshold-based versions have the same impossibility.

Unfortunately, the versions of the state-of-the-art optimistic fair exchange protocols we analyzed in Section 2.1 *without* any timeouts fall under this protocol category. Note that, this means, using static resolution sets and autonomous arbiters, those protocols cannot be extended to use multiple arbiters and remain fair.

[9] This corresponds to the familiar "static corruption model" in many other works.

# 6 Relaxing Autonomous Arbiters Assumption

In this section, we extend our framework by relaxing the autonomous arbiters assumption to allow for ordered aborts by the honest party and therefore include a broader range of protocols in our framework. We still assume that the honest arbiters do not try to communicate, but now the honest parties can contact the arbiters following some particular order. We immediately notice that the only places where we need that assumption are Scenario 2 and Protocol 2. Results about all other scenarios and protocols stay unchanged when we do the relaxation by removing the explicit *Dispute* state in the ITM definitions of the honest participants (Alice and Bob), thus allowing them to contact the arbiters with some specific order. Yet, we still are not considering byzantine fault tolerance or secure multiparty computation techniques.

## 6.1 Scenario 2 Revisited

In Section 4.2, we analyzed the scenario in which the malicious party has the ability to resolve only, whereas the honest party can abort and resolve. We analyzed that scenario using the autonomous arbiters assumption. Below, we will remove the requirement that arbiters are contacted simultaneously, and revisit our analysis.

### 6.1.1 Scenario 2 with Threshold-based Mechanisms

Here, we are limiting our protocol instances to the case where only threshold-based mechanisms are used.[10] This means, the sets $\mathcal{H}_{\mathcal{R}}(t)$ and $\mathcal{M}_{\mathcal{R}}(t)$ are of the specific form we have described before. Remember, the set $\mathcal{H}_{\mathcal{R}}(t)$ is composed of all subsets of $N$ with $T_H$ or more arbiters. Similarly, the set $\mathcal{M}_{\mathcal{R}}(t)$ is composed of all subsets of $N$ with $T_M$ or more arbiters. $T_H$ and $T_M$ are the corresponding thresholds.

**Lemma 5.** *Every DAFE protocol instance needs to make sure there exists a time $t$ when $T_H \leq T_M - |F(t)|$.*

*Proof.* Assume otherwise: At all times $T_H > T_M - |F(t)|$. Malicious party can break fairness as follows: When $H$ wants to abort the protocol (as directed by the protocol, most probably triggered by an incorrect input from the malicious party), $M$ waits until $H$ aborts with $n - T_H + 1$ arbiters. $H$ can no longer resolve after this point since there are less than $T_H$ arbiters left in the set of available arbiters $R_A$. At this point, $M$ intercepts any more abort messages from $H$ and resolves with $T_M - |F(t)|$ honest arbiters (as well as $|F(t)|$ friends). Therefore, this protocol instance is unfair (does not satisfy semantic fairness). □

Notice that Lemma 5 is the same as Corollary 3. Therefore, Corollaries 4 and 5 also apply here.

### 6.1.2 Scenario 2 General Case

Now, we remove all the restrictions we made on our scenario in the previous sub-scenarios. This means, we allow for any set-based resolution mechanism, and we even allow the protocol to specify an order of arbiters for aborting, possibly depending on the execution of the protocol instance. One can think of it as the honest party aborting with one arbiter at every time step, and reconsidering his decision to abort each time. Therefore, the arbiters are no longer completely autonomous.

[10] Appendix 6.1.2 removes the threshold limitation and allows for any set-based resolution mechanism.

**Lemma 6.** *Every DAFE protocol instance needs to make sure that at all times $t$ $\forall M_R \in \mathcal{M}_\mathcal{R}(t)$ $M_R \not\subseteq F(t)$ (before $H$ has resolved) AND there exists a time $t$ when $\exists H_R \in \mathcal{H}_\mathcal{R}(t)$* s.t. *$H_R \cap F(t) = \varnothing$.*

*Proof.* Assume there exists a time when $\exists M_R \in \mathcal{M}_\mathcal{R}(t)$ $M_R \not\subseteq F(t)$ (before $H$ has resolved). Malicious party can break fairness as follows: When $H$ wants to abort the protocol, $M$ lets him abort with all the arbiters. Then, he goes and resolves with $M_R$, all members of which are his friends.

Now assume at all times $\forall H_R \in \mathcal{H}_\mathcal{R}(t)$ $H_R \cap F(t) \neq \varnothing$. Malicious party can break fairness by just resolving with any $M_R \in \mathcal{M}_\mathcal{R}(t)$. Since all possible ways to resolve for $H$ has to go through one of the malicious party's friends, he has no hope of resolving. □

In this general scenario, as in the previous cases, we would like to be able to prove that any DAFE protocol instance needs to make sure there exists a time $t$ when $\forall M_R \in \mathcal{M}_\mathcal{R}(t)$ $\exists H_R \in \mathcal{H}_\mathcal{R}(t)$ *s.t.* $H_R \subseteq M_R - F(t)$. Even though this seems a very plausible and realistic conclusion, several problems arise with its proof.

The general idea is to use an adversary very similar to the one in Section 4.2. So, the adversary will let $H$ to abort with any arbiter in $R_A - M_R$. Then, if $H$ wants to abort with an arbiter in $M_R - F(t)$, $M$ will intercept and resolve with $M_R$. The problem is that this works depending on the order of aborts. There might be a possible protocol construction and order specification that makes sure $H$ can still resolve once he detects this behavior. We do not know of and could not come up with such a construction, due mostly to the fact that $F(t)$ is unknown to the honest party, and hence designing a protocol instance using an order that works without knowing $F(t)$ seems impossible. Even though the order may work for some protocol instances, having an order that works with high probability (that works on all but negligible fraction of protocol instances) does not seem possible. Furthermore, the moment we allow for more powerful adversaries, since the order of arbiters for the honest participant to abort is public, the adversary might "bribe" some "key" arbiters to become his friends and make sure the ordering fails to provide fairness (in the dynamic/adaptive corruption model). We admit that we have no proof for this general case with less powerful adversaries, but we conjecture that the same predicate for scenario 4.2 as before will hold.

## 6.2 Protocol 2 Revisited (More Impossibility Results)

In this type of protocols, Alice is given the ability to abort and resolve, whereas Bob is given only the ability to resolve. Analysis of protocols that are symmetric to this type of protocols (where Bob can abort and resolve, and Alice can only resolve) obviously yields to the same conclusions. The predicate for case 1 changes when we relax our autonomous arbiters assumption. Case 2 stays the same. Remember, the resolution sets we consider here are static.

**Case 1: Honest Alice *vs.* Malicious Bob**: This case falls under Scenario 2, which requires special treatment when arbiters are not contacted simultaneously for aborting. For threshold-based mechanisms, every DAFE protocol needs to have $T_A \leq T_B - |F_B|$. For the most general case of DAFE protocols, we need $\forall B_R \in \mathcal{B}_\mathcal{R}$ $B_R \not\subseteq F_B$ AND $\exists A_R \in \mathcal{A}_\mathcal{R}$ *s.t.* $A_R \cap F_B = \varnothing$ (see Lemma 6 in Appendix 6.1.2).

**Case 2: Malicious Alice *vs.* Honest Bob**: This case falls under Scenario 1, which means any DAFE protocol needs to have $\forall A_R \in \mathcal{A}_\mathcal{R}$ $\exists B_R \in \mathcal{B}_\mathcal{R}$ *s.t.* $B_R \subseteq A_R - F_A$. Remember, Corollary 3 (using threshold-based mechanisms) require $T_B \leq T_A - |F_A|$.

Regarding DAFE protocols using threshold-based arbiter resolution mechanisms, we can conclude (from the two cases above) that no such meaningful protocol can exist ($T_A \leq T_B - |F_B|$ and $T_B \leq T_A - |F_A|$ gives $T_A \leq T_A - |F_A| - |F_B|$, which means all the arbiters need to be trusted). Hence, there is no point in distributing the arbiters in terms of trust. It is even worse since we need to trust every single arbiter, and the protocol cannot be fair even if only one arbiter is corrupt.

Regarding general set-based DAFE protocols, we cannot conclude an immediate impossibility. But following our discussion above, we conjecture that no such useful protocol can exist.

Unfortunately, as we have shown in Section 2.1, the versions of the state-of-the-art protocols we analyzed in Section 2.1 *without* any timeouts fall under this protocol category. So the impossibility with threshold-based mechanisms, and our conjecture apply to very common real cases, even when the arbiters are not contacted simultaneously by the honest party.

# 7 Applying DAFET Framework to Prove Optimality of an Existing Protocol

In this section, we analyze an existing DAFET protocol that uses dynamic resolution sets: The set of arbiters needed by a party for resolution changes during the course of the execution of the protocol instance. By adjusting resolution sets reactively, this protocol can provide semantic fairness.

**AV Protocol [6]** This protocol is due to Avoine and Vaudenay (AV) [6]. In this protocol, timeouts are used for aborting (it is a DAFET protocol). It is a three-step protocol in which Alice starts by sending verifiable secret shares encrypted under each arbiter's public key. Then, Bob responds with his secret, and Alice responds with her secret. To resolve, Bob contacts $k$ arbiters to get the decrypted shares and reconstruct the secret of Alice (where $k$ is the threshold for the secret sharing scheme). Before giving the decrypted share, each honest arbiter asks for the secret of Bob.[11] Hence, the set $\mathcal{B}_{\mathcal{R}}(t)$ contains all subsets of $N$ with $k$ or more arbiters and $\mathcal{A}_{\mathcal{R}}(t)$ is initially empty[12].

The state semantics obviously coincide with our 3-state definition. The participants either succeed in obtaining the other party's exchange item and hence end at their *Resolved* state, or they fail to do so and end at their *Aborted* state. The honest arbiters will either help both participants, or abort at the timeout and help neither.

Even though in the AV protocol the honest arbiters directly contact Alice when Bob resolves with them, we can see it as the arbiters storing Bob's secret, and Alice contacting them to obtain Bob's secret later on. Since Alice can only resolve after Bob, and Bob has to resolve before the timeout, it is safe to think of this protocol as letting Alice to resolve only after the timeout. Unlike the protocols in Section 5 which were proven impossible to be fair, this protocol uses dynamic resolution sets that help it achieve fairness (we talk about the relationship between timeouts and dynamic resolution sets in Section 8). So, sets $\mathcal{H}_{\mathcal{R}}(t)$ and $\mathcal{M}_{\mathcal{R}}(t)$ change according to the following additional rule regarding the actions (remember the actions in Section 3):

**Action 6** (Bob resolves with an arbiter $X \in R_A$)**.** *The effect is that a set* $\{X\}$ *is added to the set of sets* $\mathcal{A}_{\mathcal{R}}(t)$*.*

[11] The user should refer to [6] for any more details.

[12] It does not contain the empty set, it is empty. This means no set of arbiters is sufficient for Alice to resolve.

This rule is there since in the AV protocol, when Bob contacts an honest arbiter, that arbiter contacts Alice and sends Bob's whole secret. It guarantees that the moment a malicious Bob resolves with any honest arbiter, Alice is guaranteed to be able to resolve. Let us analyze the two cases and see how this protocol satisfies the lemmas regarding scenarios.

**Case 1: Honest Alice *vs.* Malicious Bob**: This case falls under Scenario 3, which means any DAFET protocol needs to make sure there exists a time when $\forall B_R \in \mathcal{B}_{\mathcal{R}}(t)\ \exists A_R \in \mathcal{A}_{\mathcal{R}}(t)$ *s.t.* $A_R \subseteq B_R - F_B$.

**Case 2: Malicious Alice *vs.* Honest Bob**: Depending on at which point of the protocol the resolution begins, malicious Alice might have already resolved, thus this case falls under Scenario 4, which requires that there exists a time when $\exists B_R \in \mathcal{B}_{\mathcal{R}}(t)$ *s.t.* $B_R \cap F_A = \varnothing$.

**Lemma 7.** *AV protocol cannot provide semantic fairness unless for all times* $t$ $\forall B_R \in \mathcal{B}_{\mathcal{R}}(t)$ $B_R \not\subseteq F_B$ *AND for some time* $t$ $\exists B_R \in \mathcal{B}_{\mathcal{R}}(t)$ s.t. $B_R \cap F_A = \varnothing$.

*Proof.* Follows from the analysis of the cases above using corollary 1 for case 1. □

The AV protocol achieves semantic fairness using dynamic sets as follows: The set $\mathcal{A}_{\mathcal{R}}(t)$ is initially empty. When Bob contacts an arbiter $X$, action 6 above takes place, and hence the set $\{X\}$ is added to the set of sets $\mathcal{A}_{\mathcal{R}}(t)$ (the threshold for Alice effectively becomes 1). Therefore, once Bob contacts an honest arbiter (not one of his friends), then Alice is guaranteed to be able to resolve. This saves an honest Alice against a malicious Bob (case 1). In case 2, as long as Bob can find a set of honest arbiters that he can resolve with, he is saved against malicious Alice.

Actually, the AV protocol [6] uses threshold-based mechanisms instead of set-based ones, therefore we have the following corollary:

**Corollary 6.** *AV protocol cannot provide semantic fairness unless* $|F_B| < T_B$ *AND* $T_B \leq n - |F_A|$.

It is important to notice that the AV paper [6] proves essentially the same result: They prove that the same bound is also sufficient for their protocol. Thus, we have proven that the bounds proven in that paper are tight and hence the protocol is optimal in that sense. Furthermore, this result is applicable to all protocols of the same type; no DAFET protocol of the same type can achieve better bounds. In particular, the same technique of employing multiple autonomous arbiters can be used on [4] and [18] (as described in Section 2.1) to convert their timeout-based versions to DAFET protocols, and the same lemma will hold. This shows how our framework can easily be applied to prove optimality of a protocol and extended to other protocols of the same type.

As the corollary immediately reveals, when using $n$ arbiters, to obtain maximum tolerance, one should set the threshold for Bob $T_B = n/2$ so that the protocol tolerates up to $n/2 - 1$ friends of each participant. Of course, this greatly reduces the efficiency of the resolution of the optimistic fair exchange protocol.

The AV protocol needs to make the following assumption [6]: The threshold for the secret sharing scheme used for distributing the arbiters must be greater than the number of friends the malicious party can have. This limits the applicability of the protocol in real scenarios. If the threshold is set very high to tolerate worse situations, then the efficiency greatly decreases. Otherwise, if the threshold is low, than the tolerance against malicious behavior is low.

# 8 Discussion: Timeouts and Dynamic Resolution Sets

As we have proved in Section 5, no realistic DAFE protocol can provide fairness, whereas Section 7 shows an existing DAFET protocol that employs timeouts. Therefore, we can conclude that timeouts play an important role in optimistic fair exchange protocols when we would like to employ multiple autonomous arbiters. Even without completely autonomous arbiters, Section 6.2 shows an impossibility of DAFE protocols using threshold-based mechanisms, and even with set-based mechanisms, it is not clear how such a DAFE protocol can be constructed.

Timeouts are tied to the use of dynamic sets in general (as we did for DAFET protocols). When only one party can resolve before the timeout, static resolution sets lose their meaning since the resolution set for the party who cannot resolve before the timeout is empty until the timeout. That set gets defined only after the timeout, which results in that set being dynamic in a very basic sense. The dynamism prevents the adversary from coming up with a strategy that violates fairness. As shown in Section 7, this helps AV protocol achieve semantic fairness. Of course, a careful protocol design is still necessary since timeouts and dynamically changing sets by themselves do not mean that the protocol will be trivially fair. One may further argue that dynamically changing resolution sets is a more important concept that plays a big role in this (im)possibility result, but it is easy to see that timeouts are natural mechanisms to achieve this dynamism.

This suggests that even though timeouts may not be a nice feature in terms of system design, it really helps when the system needs to be extended to use multiple autonomous arbiters (together with the use of dynamically changing resolution sets).

# 9 Conclusion

In this paper, we presented a framework to analyze DAFE protocols, which are natural extensions of optimistic fair exchange protocols to make them use multiple autonomous arbiters (those who do not communicate with each other). Autonomy is useful for realistic (efficient) protocols, especially in p2p settings. Using the presented framework, we answered two open questions since [6]. We have proved that DAFE protocols (optimistic fair exchange protocols that employ multiple autonomous arbiters and does not have timeout mechanisms) cannot provide fairness in a realistic setting. Even when we extended our framework by relaxing the autonomy assumption about the arbiters, we found out that even broader classes of optimistic fair exchange protocols fall under our impossibility results. We then switched to the DAFET model to include timeouts and dynamically changing sets of arbiters to resolve with. We analyzed one existing DAFET protocol [6] using our framework and proved that the previous bounds on the required number of honest arbiters are optimal. No DAFET protocol of the same type can achieve better bounds, since our framework can easily be used to come up with generalized results. We also showed that timeouts and dynamic resolution sets play an important role in the design of such distributed arbiter fair exchange protocols.

Unfortunately, this means many optimistic fair exchange protocols that want to efficiently distribute their arbiters may need to employ synchronized clocks. And even in this case, they cannot hope to require fewer honest arbiters than the Avoine and Vaudenay protocol [6]. If they do not want to employ synchronized clocks, then they may need to employ costly solutions like secure multi-party computation or Byzantine agreement.

One may want to settle down for weaker security guarantees against weaker adversaries to achieve cheaper solutions than Byzantine agreement. Using Byzantine fault tolerance techniques

in [1], the arbiters can keep updating some value that is related to the resolution semantics of the fair exchange. Unfortunately, when aborts are considered, it is not clear if the same techniques can be applied here. We leave research in this direction as an open problem.

Finally, our techniques may be applicable to other functionalities that can be implemented using secure multi-party computation. By designing an appropriate framework, we may prove more general results about achieving the same functionality using autonomous multiple parties. We leave such a generalization as an interesting open problem.

## References


[1] M. Abd-El-Malek, G. Ganger, G. Goodson, M. Reiter, and J. Wylie. Fault-scalable byzantine fault-tolerant services. In *SOSP*, 2005.

[2] N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In *ACM CCS*, 1997.

[3] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. In *EUROCRYPT*, 1998.

[4] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Selected Areas in Communications*, 18:591–610, 2000.

[5] G. Ateniese. Efficient verifiable encryption (and fair exchange) of digital signatures. In *ACM CCS*, 1999.

[6] G. Avoine and S. Vaudenay. Optimistic fair exchange based on publicly verifiable secret sharing. *ACISP*, 2004.

[7] F. Bao, R. Deng, and W. Mao. Efficient and practical fair exchange protocols with off-line TTP. In *IEEE Security and Privacy*, 1998.

[8] M. Belenkiy, M. Chase, C. Erway, J. Jannotti, A. Küpçü, A. Lysyanskaya, and E. Rachlin. Making p2p accountable without losing privacy. In *ACM WPES*, 2007.

[9] M. Ben-Or, R. Canetti, and O. Goldreich. Asynchronous secure computation. In *STOC*, pages 52–61, 1993.

[10] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *STOC*, 1988.

[11] J. Camenisch and I. Damgård. Verifiable encryption, group encryption, and their applications to group signatures and signature sharing schemes. In *ASIACRYPT*, 2000.

[12] J. Camenisch, A. Lysyanskaya, and M. Meyerovich. Endorsed e-cash. In *IEEE Security and Privacy*, 2007.

[13] J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *CRYPTO*, 2003.

[14] R. Canetti and T. Rabin. Fast asynchronous byzantine agreement with optimal resilience. In *STOC*, 1993.

[15] Y. Dodis, P. Lee, and D. Yum. Optimistic fair exchange in a multi-user setting. In *PKC*, 2007.

[16] E. Fujisaki and T. Okamoto. A practical and provably secure scheme for publicly verifiable secret sharing and its applications. In *EUROCRYPT*, volume 1403 of *LNCS*, pages 32–46, 1998.

[17] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC*, pages 218–229, 1987.

[18] A. Küpçü and A. Lysyanskaya. Usable optimistic fair exchange. In *CT-RSA*, 2010.

[19] S. Micali. Simultaneous electronic transactions with visible trusted parties. US Patent 5,553,145, 1996.

[20] S. Micali. Simple and fast optimistic protocols for fair electronic exchange. In *PODC*, 2003.

[21] H. Pagnia and F. Gärtner. On the impossibility of fair exchange without a trusted third party. *Darmstadt University of Technology*, TUD-BS-1999-02, 1999.

[22] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, 1991.

# A Definition of Optimistic Fair Exchange Protocols

We provide an informal definition of optimistic fair exchange protocols taken from [18] just for completeness.

A fair exchange protocol is composed of three interactive algorithms: Alice running algorithm $A$, Bob running algorithm $B$, and the Arbiter running the trusted algorithm $T$. Alice has content $f_A$, and Bob has content $f_B$

Completeness for an optimistic fair exchange states that the interactive run of $A$ and $B$ by *honest parties* results in $A$ getting $f_B$ and $B$ getting $f_A$ (the Arbiter's algorithm $T$ is not involved, assuming an ideal network).

Fairness states that at the end of the protocol, either Alice and Bob both get the content of each other, or neither Alice nor Bob gets anything useful. For formal definitions, we refer the reader to [18].