Fairness in the Wild: Secure Atomic Swap with External Incentives

Hao Chung*
Carnegie Mellon University
chunghaoqc@gmail.com

Elisaweta Masserova†
Carnegie Mellon University
elisawem@andrew.cmu.edu

Elaine Shi*
Carnegie Mellon University, Oblivious Labs Inc., and 0xPARC
elainershi@gmail.com

Sri AravindaKrishnan Thyagarajan*
University of Sydney
aravind.thyagarajan@sydney.edu.au

Abstract

Atomic swaps enable asset exchanges across blockchains without relying on trusted intermediaries, and are a key component of decentralized finance (DeFi) ecosystems. Recently, Chung, Masserova, Shi, and Thyagarajan introduced *Rapidash* (Financial Cryptography 2025), an atomic swap protocol that remains incentive compatible under user-miner collusion, by ensuring that the honest strategy forms a coalition-resistant Nash equilibrium. However, their model assumes a closed system where players act solely based on internal protocol incentives. In practice, participants may be influenced by *external incentives* such as off-chain rewards or adversarial bribes, which can undermine such equilibrium guarantees.

In this work, we introduce a new game-theoretic notion, *bounded maximin fairness*, which ensures that honest participants remain protected against rational adversaries with arbitrary but bounded external incentives. We construct an atomic swap protocol that satisfies this notion, while preserving the equilibrium properties of prior work in the absence of external influence. As we show, our protocol is easy to implement and can be instantiated even in Bitcoin's limited scripting language.

*Supported by NSF awards 2212746, 2044679, 1704788, a Packard Fellowship, a generous gift from the late Nikolai Mushegian, a gift from Google, and an ACE center grant from Algorand Foundation.

†Supported by NSF Grant 1801369, the CONIX Research Center, the Defense Advanced Research Projects Agency under contract FA8750-17-1-0059, a gift from Bosch, and the CBI postdoctoral fellowship.

Contents

1 Introduction

In distributed systems and cryptographic protocols, fairness is a central yet elusive goal. However, achieving the strongest form of fairness — either all parties receive the output or none do — is often impossible for general multi-party functionalities without heavy assumptions such as honest majority or trusted execution environments [PG99]. This limitation has motivated the search for relaxed fairness notions that still provide meaningful guarantees for honest participants, even when full fairness is unattainable.

One promising direction is to embrace a game-theoretic perspective, recognizing that real-world participants are often *rational* parties, motivated by self-interest rather than classified as either honest or malicious in the classical cryptographic sense. Protocols in this paradigm aim to ensure that no coalition of rational parties has an incentive to deviate from the prescribed behavior. This shift in perspective has already led to substantial progress in domains previously blocked by impossibility results. For instance, in the case of fair coin tossing recent game-theoretic protocols [WAS22, TSW24, CGL+18] obtained fairness even under dishonest majority — a setting where classical cryptographic notions of fairness have long been known to be impossible to achieve.

We revisit another fundamental primitive in decentralized environments: atomic swaps. These protocols enable two mutually distrustful parties to exchange digital assets across different chains without relying on a trusted intermediary. However, the blockchain setting introduces new challenges: miners can selectively include, exclude, or reorder transactions, and users may collude with the miners to extract unfair advantage. To address this, a recent work dubbed Rapidash [CMST22b] laid a foundational game-theoretic framework for atomic swaps that remain incentive-compatible even in the presence of user-miner collusion. The Rapidash protocol was proven to satisfy cooperative strategy proofness (CSP-fairness), which guarantees that the coalition's utility is maximized when all players in the coalition follow the honest strategy. Thus, informally, in Rapidash's atomic swap the honest strategy formed a coalition-resistant Nash equilibrium.

Yet, Rapidash left one important question open: what if parties have access to external incentives, such as side payments or off-chain contracts with external entities? In realistic settings, particularly in decentralized finance (DeFi), participants do not operate in isolation. They may be subject to off-protocol pressures or rewards that alter their incentives. Consider an atomic swap where one party, Alice, is a major liquidity provider on a decentralized exchange, and her ability to maintain this role depends on successfully completing a swap with Bob, e.g., acquiring BTC to rebalance her position. Suppose a third party, Eve, stands to gain market dominance if Alice's trading capacity is disrupted. In this scenario, Eve may have a strong incentive to bribe Bob to misbehave during the atomic swap — ensuring that Alice does not receive the BTC in time, thereby impairing her operations. Note that in this case, the CSP guarantee is insufficient, as it offers no protection when Bob is willing to adopt a strategy that does not maximize his utility within the swap itself. Without external incentives, such scenarios are not of concern, since it is reasonable to assume that Bob would always aim to maximize his own profit within the atomic swap. However, in the presence of off-protocol incentives, Bob may find it optimal to even intentionally *lose money* within the atomic swap protocol as long as Eve's compensation offsets this loss.

As out-of-band incentives clearly violate the closed-world assumption underlying many rational protocol models and can undermine the guarantees of even game-theoretically secure constructions, in this work we ask ourselves the following question:

Is it possible to provide an atomic swap protocol that is secure against strategic players with external incentives?

1.1 Our Contributions

We answer this question positively, and our contribution here is two-fold. First, we formalize a new game-theoretic notion, called *bounded maximin fairness*, which captures the idea that honest players should not be harmed by the presence of external incentives, and it may be of independent interest in a broader context. Second, we design an atomic swap protocol that satisfies this notion. Moreover, we present the Bitcoin instantiation of our protocol showcasing the practicality of our solution.

New definition: bounded maximin fairness. As explained above, the notion of CSP fairness does not capture scenarios where parties can be influenced by external incentives. Intutively, when such incentives are unbounded, they can motivate the participating parties to behave arbitrarily maliciously. In this sense, the strongest possible game-theoretic notion one can hope for is that even when the strategic coalition can be arbitrarily malicious, honest players should not be harmed. This idea underpins *maximin fairness*, a notion introduced in recent works [PS17,CGL+18,WAS22]. Maximin fairness implicitly assumes that the external incentives may be arbitrary and unbounded, which explains why the strategic players' behavior may appear arbitrarily malicious from the perspective of the present protocol. Maximin fairness, however, is a very stringent requirement, and insisting on such a strong notion of fairness may severely constrain the design space or even lead to impossibility results in some applications.

For atomic swap, we currently do not know how to achieve maximin fairness given strategic miners. Instead, we suggest a meaningful relaxation called *bounded maximin fairness*. Unlike its more stringent counterpart, bounded maximin fairness allows external incentives to be arbitrarily large but assumes *there is an upper bound on the external incentives and the bound is known to the protocol*. We want to guarantee that honest participation will not lead to negative utility as long as the other strategic players behave rationally in light of the arbitrary but bounded external incentives. This guarantee nicely complements equilibrium-type guarantees such as CSP fairness — essentially, bounded maximin provides a safety net for honest parties if they suspect that there may be external influence at play. In other words, simple-minded, non-strategic players can always feel safe participating in the protocol.

Secure atomic swap. We design an atomic swap protocol that satisfies both bounded maximin fairness and CSP fairness. Additionally, our protocol has *dropout resilience*, which ensures that an honest player is protected from loss if the other player plays honestly, but drops offline at any point during the protocol. This property is important in practice since the honest player may go offline due to loss of password or network outage.

Finally, we implement our protocol for blockchains that support general smart contracts, and also demonstrate how it can be instantiated with Bitcoin scripts — showcasing its practicality even on more restrictive platforms like Bitcoin.

1.2 Related Work

Over the past decades, significant effort has been devoted to achieving fair exchange on blockchains. Numerous elegant solutions have been proposed under the assumption that miners behave honestly. For example, Choudhuri et al. [CGJ+17] presented a solution for achieving fair exchange and fair multi-party computation using a public bulletin board and extractable witness encryption. For atomic swaps, several approaches have been developed [Her18, MMS+, vdM19, MD19, ZDBN19]. Typically, atomic swaps are based on hashed time-lock contracts (HTLCs) [Her18] or a novel cryptographic primitive known as *adaptor signatures* [DH20].

Another line of work focuses on achieving fair exchange when miners may behave strategically [CMST22b, WSZN22, CMST22a]. A series of papers [WHF19, Bon16, MHM18] demonstrated how users can bribe miners to censor competing transactions. To improve the robustness of HTLCs under such adversarial conditions, Tsabary et al. [TYME21] proposed *MAD-HTLC*, which requires the initiating party to lock up extra collateral. Although MAD-HTLC mitigates certain bribery attacks, it fails to achieve CSP-fairness in the presence of general user-miner collusion. More recent works, such as Rapidash [CMST22b] and He-HTLC [WSZN22], extend the ideas behind MAD-HTLC. These works propose new *knowledge-coin exchange* protocols, which achieve the same functionality as HTLCs, while providing provable CSP-fairness guarantees. However, none of these protocols satisfies the bounded maximin fairness.

Interestingly, as pointed out by Chung et al. [CMST22b], even if the underlying knowledge-coin exchange protocols are CSP-fair, the atomic swap protocol built upon them may not inherit this property. This highlights that game-theoretic properties like CSP-fairness are not necessarily composable. Therefore, a carefully designed atomic swap protocol is required to ensure CSP-fairness. A more detailed discussion on achieving CSP-fairness in atomic swaps is provided in Section 2.2.

Beyond fair exchange, a notable category of external incentives in blockchain systems is *oracle manipulation* attacks. An oracle is a service that provides real-world data to smart contracts on a blockchain. Some oracles are themselves implemented as smart contracts, where users stake tokens and receive rewards for supplying data. When the outcomes of other applications, e.g., prediction markets, depend on the oracle's data, these applications introduce external incentives that may motivate oracle participants to submit incorrect information in order to manipulate outcomes. It was reported that over $400 million was lost to oracle manipulation attacks in 2022 [ora22].

2 Technical Overview

The goal of an atomic swap is to enable two parties to exchange their cryptocurrencies across two different blockchains. Before diving into our solution, we first introduce a helpful simpler primitive, *knowledge-coin exchange*. In Section 2.1, we review the existing knowledge-coin exchange protocols, and explain why they fail to achieve bounded maximin fairness even with limited external incentives. Then, we explain the necessary modifications to achieve bounded maximin fairness. In Section 2.2, we explain how to realize atomic swap based on knowledge-coin exchange. We highlight the key challenges of designing an atomic swap protocol, and summarize the key ideas of our construction. In the following, we use "strategic players," or "coalition," to describe colluding parties (that can potentially exhibit malicious behavior).

2.1 Knowledge-Coin Exchange

Hash timelock contract (HTLC). In a knowledge-coin exchange, Bob wants to buy a secret s from Alice at a price of $\$v$ coins. In practice, such functionality is often realized by *hash timelock contracts* (HTLCs), which work as follows. Alice first sends $h_s = H(s)$ to Bob, where $H(\cdot)$ denotes a cryptographic hash function. Then, Bob deposits $\$v$ into the contract upfront, and the contract is parametrized with the hash value h_s and a timeout T.

HTLC

- On receiving s from Alice such that $H(s) = h_s$, send $\$v$ to Alice.

- After T, on receiving ping from Bob, send $\$v$ to Bob.

Above, ping is a predetermined message which triggers the corresponding *activation point*, i.e., the corresponding part of the smart contract logic. The first activation point allows Alice and Bob to exchange the secret and the payment, and the second one allows Bob to recover its deposit if Alice drops offline. Since the total deposit is only $\$v$, the two activation points are mutually exclusive, i.e., only one of them can be activated and only once.

While HTLC is dropout resilient, it is unfortunately not CSP fair in the presence of user-miner coalitions. If Alice offers a transaction fee of $\$f$, then as long as Bob bribes each miner $\$f + \$\epsilon$ (for some small $\epsilon > 0$) for excluding Alice's transaction, rational miners will take the bribe [TYME21]. This bribery attack makes sense for Bob as long as $\$v > \$f \cdot T$. It may seem like HTLC is secure as long as $T \cdot \$f$ is sufficiently large. However, Tsabary et al. [TYME21] showed new attacks where the cost to Bob is not dependent on T.

Prior work: Achieving CSP-fairness. To address user-miner collusion, Chung et al. [CMST22b] proposed a new protocol that achieves CSP-fairness. In this protocol, Alice first computes $h_s = H(s)$. Additionally, Bob samples a random string pre_b, and computes $h_B = H(pre_b)$. They deploy the following smart contract parametrized by hash values h_s and h_b, the timeout T_1 and T_2, the value of the secret $\$v$, the collateral $\$c_b$, and a predetermined small amount $\$\epsilon$.

CSP-fair knowledge-coin exchange contract

/* *Params:* $(h_s, h_b, T_1, T_2, \$v, \$c_b, \$\epsilon)$, *Bob deposits* $\$v + \$c_b$. */

P_{default}: On receiving z from Alice s.t. $H(z) = h_s$, send $\$v$ to Alice and $\$c_b$ to Bob.

P_{refund}: Time T_1 or greater: on receiving z from Bob s.t. $H(z) = h_b$, do nothing.

C_{refund}: At least T_2 after P_{refund} is activated: on receiving ping from anyone, send $\$v + \$c_b$ to Bob.

C_{burn}: On receiving (z_1, z_2) from any P s.t. $H(z_1) = h_s$ and $H(z_2) = h_b$, send $\$\epsilon$ to player P. All remaining coins are burnt.

Figure 1: Rapidash's CSP-fair knowledge-coin exchange contract [CMST22b].

In the above construction, the activation points starting with the same capital letter are mutually exclusive. For example, once P_{default} is activated, P_{refund} cannot be activated. We adopt the same convention in this work. The formal specification of writing a smart contract is given in Section 3.1.

To complete the knowledge-coin exchange, Bob deposits the intended payment $\$v$, and an additional $\$c_b$ amount of collateral into the contract. Then, Alice sends s to P_{default} as soon as Bob's deposit takes effect. In this case, Alice receives the payment $\$v$, and Bob learns s and gets his collateral back all at once. If Alice fails to post s by time T_1, Bob sends pre_b to P_{refund} at time T_1 to request a refund. The activation point P_{refund} merely allows Bob to express his intent to request a refund. The actual refund happens when T_2 time has passed since the activation of P_{refund} — at this point, Bob sends to C_{refund} which actually sends him the refund. An honest miner always includes all outstanding transactions in any block it mines. Importantly, if an honest miner has observed both s and pre_b contained in the transactions posted, it will immediately post (s, pre_b) to C_{burn}, and this transaction will always be ordered in front of others in the block it mines.

The knowledge-coin protocol above achieves CSP-fairness. The key insight here is that the activation point C_{burn} serves as a "bomb". Suppose that the honest Alice has posted s. Now, should a strategic Bob-miner coalition post pre_b to P_{refund} in an attempt to get refunded and thus

get the secret s for free, both s and pre_b will be publicly known. Note that the coalition has to wait at least T_2 amount of time before the actual refund C_{refund} can be activated. During this T_2 window, if any non-colluding miner mines a block, it will trigger the bomb by posting (s, pre_b) to C_{burn}, which will cause Bob to lose its collateral. Thus, if a Bob-miner coalition wishes to get the secret for free, it has to take a gamble that it will be able to mine all blocks in the T_2 window.

Why Rapidash does not satisfy bounded maximin fairness. Unfortunately, the above protocol does not satisfy bounded maximin fairness, as an externally motivated Alice can harm Bob. Indeed, Alice can withhold s and wait until Bob sends pre_b to P_{refund}. Then, Alice can send s and pre_b to C_{burn} to redeem $\$\epsilon$, thus making Bob lose his collateral $\$c_b$. As long as Alice is compensated externally by $\$v - \$\epsilon$, she is incentivized to trigger C_{burn} instead of sending s to P_{default} to earn $\$v$.

Bounded maximin fairness: definition subtleties. Before we explain our modifications to the above protocol towards one that is better aligned with our bounded maximin fairness notion, we first discuss the proposed fairness definition itself. At a first glance, a natural way to define bounded maximin fairness is to require that the *optimal strategy* that maximizes the externally incentivized coalition's utility does not harm any honest player. It implicitly assumes that the coalition *knows* the optimal strategy. However, the external incentive can be an arbitrary function of the blockchain states, and finding the optimal strategy may not be computationally efficient. To capture the security that defends against probabilistic polynomial-time (PPT) players, we define a set of *blatantly irrational strategies* $\bar{\mathcal{R}}$, where a strategy S is in $\bar{\mathcal{R}}$, i.e., it is blatantly irrational, if one can efficiently find another strategy S' such that the coalition's utility when adopting S' is strictly larger than that of S. Then, we say that a protocol is bounded maximin fair if honest players' utility is non-negative when the coalition adopts any strategy outside $\bar{\mathcal{R}}$. We provide the formal definition in Section 3.3.1.

Our knowledge-coin exchange. The challenge in designing a bounded-maximin-fair protocol is to make sure *every strategy that may harm the honest players is blatantly irrational.* We achieve this by ensuring that the following conditions hold:

1. Harming the honest player necessarily puts the adversary at risk of triggering the bomb.
2. The utility of the externally incentived coalition strictly decreases if the bomb activation point gets triggered.
3. The last message that contributes to the bomb getting triggered is always being sent by a strategic player.

The last two conditions ensure that activating the bomb is a blatantly irrational strategy: Whenever a strategic player is about to send a message which would trigger a bomb, their utility is strictly improved if they instead simply stop sending messages from that moment on. Therefore, given any strategy S, we only need to check whether stopping to send messages is a better strategy than continuing with S, which is efficiently verifiable.

To ensure that the second condition holds, we set the collateral high enough to ensure that the external incentive cannot offset the loss which occurs when a bomb is triggered. As we know that an externally incentivized Alice can harm Bob, we have Alice supply the collateral as well (and burn it together with Bob's when a bomb is triggered).

At a first glance, the second condition may seem trivial, because how and why would honest players trigger the bomb? However, surprisingly, the CSP-fair contract specified in Figure 1 does not satisfy this condition. Consider the following: A strategic Bob posts pre_b publicly even "before

making his deposit". Then, once Bob made his deposit, the honest Alice follows the protocol and posts s to P_{default}. At this moment, both s and pre_b are publicly known, so the miner can post (s, pre_b) to C_{burn} to trigger the bomb.

To prevent the scenario above, we regulate the timing when the strategic Bob would want to reveal pre_b. To do so, we add the following activation points, where $h_b = H(pre_b)$.

B_{defuse}: On receiving ping from anyone, do nothing.

B_{burn}: On receiving z from anyone P such that $H(z) = h_b$, send $\$\epsilon$ to player P and burn remaining coins.

The effect of B_{burn} is similar to C_{burn} as it burns most of the deposits, but B_{burn} can be triggered simply by pre_b. Recall that the activation points of the same type are mutually exclusive. Once B_{defuse} has been activated, B_{burn} cannot be activated. Thus, the purpose of B_{defuse} is to "defuse" the bomb of B_{burn}, i.e., a strategic Bob would only want to post pre_b *after* B_{defuse} has been activated. Since B_{defuse} can only be triggered when the deposits have been made, this ensures that a strategic Bob would not want to post pre_b before making his deposit (which was the problem in the scenario considered above). Intuitively, posting pre_b before the deposits is now a blatantly irrational strategy. Consequently, the resulting contract is as follows.

Our knowledge-coin exchange contract

/* *Params:* $(h_s, h_b, T_1, T_2, \$v, \$c_b, \$\epsilon)$*, Bob deposits* $\$v + \$c_b$*, Alice deposits* $\$c_a$*.* */

B_{defuse}: On receiving ping from anyone, do nothing.

B_{burn}: On receiving z from anyone P such that $H(z) = h_b$, send $\$\epsilon$ to player P and burn remaining coins.

P_{default}: On receiving z from Alice s.t. $H(z) = h_s$, send $\$v + \$c_a$ to Alice and $\$c_b$ to Bob.

P_{refund}: Time T_1 or greater: on receiving z from Bob s.t. $H(z) = h_b$, do nothing.

C_{refund}: At least T_2 after P_{refund} is activated: on receiving ping from anyone, send $\$c_a$ to Alice and send $\$v + \$c_b$ to Bob.

C_{burn}: On receiving (z_1, z_2) from any P s.t. $H(z_1) = h_s$ and $H(z_2) = h_b$, send $\$\epsilon$ to player P. All remaining coins are burnt.

2.2 From Knowledge-Coin Exchange to Atomic Swap

Intuitively, an atomic swap can be realized by two knowledge-coin exchange instances, one on each blockchain. Suppose Bob has x_b coins on BobChain (denoted $\ddot{\mathbb{B}}x_b$), and Alice has x_a coins on AliceChain (denoted $\ddot{\mathbb{A}}x_a$). The goal is for Bob to exchange his $\ddot{\mathbb{B}}x_b$ for Alice's $\ddot{\mathbb{A}}x_a$. The blueprint for atomic swaps via knowledge-coin exchange composition is as follows: Alice samples a random string pre_s, and sends $h_s = H(pre_s)$ to Bob. Then, she deposits a prescribed amount into a knowledge-coin exchange on AliceChain parametrized with the hash value h_s. Similarly, Bob deposits $\ddot{\mathbb{B}}x_b$ into another knowledge-coin exchange on BobChain parametrized with *the same hash value* h_s. Once both Alice and Bob have deposited their coins, Alice sends pre_s to the contract on BobChain to obtain $\ddot{\mathbb{B}}x_b$. At this moment, pre_s is publicly known, and Bob can send pre_s to the contract on AliceChain to obtain $\ddot{\mathbb{A}}x_a$.

Naive composition. We can hope that this logic can be used to satisfy bounded maximin and CSP fairness, if we use our knowledge-coin exchange scheme from above as a base. Towards

this, in addition to pre_s, Alice samples a random string pre_a to help with the refund procedure. Similarly, Bob samples a random pre_b. They deploy a contract parametrized with the hash values h_s, $h_a = H(pre_a)$ on AliceChain, and another contract parametrized with h_s, $h_b = H(pre_b)$ on BobChain. They also put the prescribed deposits and collaterals into the contracts. Ideally, when everyone is honest, Alice will send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ on BobChain (henceforth, the superscript B indicates that the activation point is in the contract on BobChain), and Bob uses the same pre_s to activate $P^{\mathsf{A}}_{\mathsf{default}}$ on AliceChain (similarly, the superscript A indicates that $P^{\mathsf{A}}_{\mathsf{default}}$ is on AliceChain) to complete the atomic swap. If the other party drops out, Alice and Bob can use pre_a and pre_b to get refunded, respectively.

However, as Chung et al. [CMST22b] pointed out, the direct composition of two CSP-fair knowledge-coin exchanges does not yield a CSP-fair atomic swap. Here we briefly review the issues and their solution, as it is crucial to our construction. The issue is that Alice-miner coalition can withhold pre_s, and try to get refunded on AliceChain. Of course, at this moment, the honest Bob will also try to get refunded on BobChain by sending pre_b, but Alice-miner coalition defers Bob's refunding transaction. After Alice successfully gets refunded on AliceChain, she immediately sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ to obtain Ƀx_b. Unlike the context of knowledge-coin exchange where Bob wants to buy the secret pre_s at certain price, in atomic swap, pre_s is just a random string facilitating the exchange. Once Alice gets refunded on AliceChain, pre_s is worth nothing to Bob. Consequently, Alice-miner coalition can get Ƀx_b for free!

To address this issue, we allow the bomb $C^{\mathsf{A}}_{\mathsf{burn}}$ on AliceChain to be triggered by pre_a and pre_b. Consequently, if Alice-miner coalition withholds pre_s so that the honest Bob tries to get refunded by pre_b, Alice-miner coalition is disincentivized to send pre_a. This patch indeed fixes the attack above, while we lose the dropout resilience for Alice. If Alice's deposit on AliceChain is delayed, e.g., due to a network congestion, Bob will send pre_b to BobChain to get refunded. At this moment, after the deposit is finalized, the honest Alice could not get refunded by pre_a because the bomb $C^{\mathsf{A}}_{\mathsf{burn}}$ can be triggered by (pre_a, pre_b).

A key challenge of designing an atomic swap protocol is to find the right balance for how easy the player can be refunded. If it is too easy, it may become risk-free to attack the honest player; if it is too difficult, the honest player's coins may be locked simply due to network congestion.

Two-phase preparation. Chung et al. [CMST22b] proposed a two-phase preparation to address the above issue. They locked $P^{\mathsf{B}}_{\mathsf{default}}$ and $C^{\mathsf{B}}_{\mathsf{burn}}$ on BobChain with an additional hash h_c of a random string pre_c sampled by Bob. Bob the published pre_c if the deposits into the contracts on both blockchains are finalized in a timely manner. The extra lock h_c then helped to distinguish the two cases depending on whether Alice strategically withholds pre_s or not.

1. If the deposit transactions are not finalized in time, Bob will never publish pre_c. In this case, Bob will send ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ and send pre_b to $P^{\mathsf{B}}_{\mathsf{default}}$ to get refunded. Because $P^{\mathsf{B}}_{\mathsf{default}}$ is locked with h_c, Alice cannot cash out Ƀx_b from $P^{\mathsf{B}}_{\mathsf{default}}$, and Bob can safely help Alice get refunded.

2. If the deposit transactions are finalized in time, then Bob will publish pre_c. Here, if Bob ever needs to post pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$, it must be because Alice-miner coalition withholds pre_s.

Bounded-maximin-fair atomic swap. We adapt the ideas above and combine these with our knowledge-coin exchange to achieve bounded-maximin-fair atomic swap. Specifically, Alice and Bob deploy a knowledge-coin exchange contract on AliceChain with the following modifications:

- $C^{\mathsf{A}}_{\mathsf{burn}}$ can be triggered by pre_a and pre_b. This is to prevent Alice-miner coalition from withholding pre_s on BobChain.

- Add an activation point $A^{\mathsf{A}}_{\mathsf{burn}}$:
  - On receiving z_1 from anyone P such that $H(z_1) = h_a$, or on receiving z_2 from anyone P such that $H(z_2) = h_b$, send $\ddot{A}\epsilon^{\mathsf{A}}$ to player P.

  $A^{\mathsf{A}}_{\mathsf{burn}}$ regulates the timing that Alice and Bob can reveal pre_a and pre_b, respectively, and it is crucial to the proof of bounded-maximin fairness. We also add $A^{\mathsf{A}}_{\mathsf{defuse}}$ to defuse the bomb of $A^{\mathsf{A}}_{\mathsf{burn}}$.

- $P^{\mathsf{A}}_{\mathsf{refund}}$ can be trigger by ping sent by Bob. This is to ensure that Bob can help Alice get refunded if the deposits are not finalized in time.

On BobChain, we modify the knowledge-coin exchange contract as follows:

- $C^{\mathsf{B}}_{\mathsf{burn}}$ can be triggered by pre_s, pre_b and pre_c. The revelation of pre_c indicates that the deposits into the contracts on both blockchains are finalized in a timely manner. Thus, once the honest Alice posts pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$, Bob should never try to get refunded by pre_b.

- $B^{\mathsf{B}}_{\mathsf{burn}}$ can be triggered by pre_c instead of pre_b. Because of the two-phase preparation, we need to ensure that Bob only posts pre_c after the deposits are finalized, which is crucial to the proof of bounded-maximin fairness.

- $P^{\mathsf{B}}_{\mathsf{refund}}$ can be trigger by ping sent by Alice. This step is crucial to for Alice to get refunded from BobChain if Bob drops out.

We give the complete specification of our contracts in Figure 3 and formally prove the full construction secure in Section 5.

3 Model

3.1 Blockchain and Smart Contracts

We model a *blockchain* as an append-only public ledger consisting of a sequence of ordered blocks. We consider an idealized mining process in which, at each block height t, a miner is selected with probability proportional to its mining power. This model captures both proof-of-work and proof-of-stake blockchains, where mining power corresponds to computational power and staked assets, respectively. The selected miner may then choose any subset of transactions from the pool of pending transactions and append a new block to the blockchain. Transactions in the block are executed in the order they appear.

We model a *smart contract* as an ideal functionality that can send and receive coins from players and the state of a smart contract is publicly observable. A smart contract may contain one or more *activation points*. Each transaction has a unique identifier and consists of: 1) an activation point of a smart contract, 2) a non-negative amount of coins, and 3) an arbitrary message. When a transaction is executed, the corresponding activation point of the smart contract is invoked, triggering the computation specified by the contract and possibly transferring coins. For simplicity, we use the phrase "A user sends a message msg to an activation point A" to mean that the user sends a transaction to the activation point A with message msg. The balance of a smart contract is defined as the net amount of coins it has received minus those it has sent, and must remain non-negative.

We use the following style of pseudo-code to express smart contracts. We use ping to denote an empty message.

A toy contract

- **Parameters:** time T. Alice deposits $\$d_a$, Bob deposits $\$d_b$.

A_{fast}: On receiving ping from Alice: send $\$d_b$ to Alice.
A_{wait}: After T, on receiving ping from Alice: send $\$d_a + \$d_b$ to Alice.
B_{other}: On receiving ping from Bob: send $\$d_a$ to Bob.

The capital letter in the contract defines the *type* of an activation point. All activation points of the same type are *mutually exclusive*. For example, if A_{wait} has been invoked, then neither A_{fast} nor A_{wait} can be invoked again. If an activation point is constrained to a specific time interval (e.g., after block height T), then any attempt to invoke it outside that interval is considered invalid and is ignored. An activation point cannot be invoked if the contract's balance is insufficient to cover the amount it is supposed to send. For example, if A_{wait} has been invoked, B_{other} cannot be invoked anymore.

The contract parameters specify the amounts of coins that must be deposited for the contract to become *active*. Once *all* required deposits are in place, the contract becomes active and its activation points can be invoked. Before the contract becomes active, each player is allowed to withdraw their own deposit if other players has not yet made the deposits. However, once the contract is active, the distribution of coins is only possible through the activation points.

3.2 Players and Strategy Spaces

There are three types of players in the model: Alice, Bob, and the miners. Alice and Bob are the two users who wish to exchange coins across different blockchains. We refer to them collectively as *users* to distinguish them from miners. All players are modeled as probabilistic polynomial-time (PPT) interactive Turing machines that can adaptively decide how to act based on the view in the protocol so far. We consider the following strategy spaces. Any player, including Alice, Bob, and miners, is allowed to do the following at any time:

1. **Post a transaction.** A player may post a transaction to the network at the beginning of any time step. We assume a synchronous network with zero delay. Transactions posted at time t are immediately visible to all players and miners. When miners decide which transactions to include in a block at time t, they can observe all transactions posted at that time.

2. **Create smart contracts.** A player may create an arbitrary smart contract and deposit any amount of coins into it. For example, a smart contract might specify: "If the blockchain state satisfies a particular predicate at some future time, transfer a specified amount of coins to a particular pseudonym," where both the recipient and the amount may depend on the state of the blockchain.

In addition to the above, miners are further allowed the following:

3. **Block mining.** When a miner is selected to mine a block, it may include an arbitrary subset of the outstanding transactions into the block and order them arbitrarily. The miner may also generate and include new transactions of its own.

Alice or Bob can form a coalition with some miners. Within a coalition, all private information is shared among members. The strategy space of a coalition is defined as the union of the strategy spaces of its individual members.

Throughout the paper, except in Section 6, we do not explicitly model the process of coalition formation. Instead, we assume that all members within a coalition are bound together and act to

maximize the coalition's joint utility. For simplicity, we ignore the transaction fee in our model. The bribery attack done by the transaction fee can be viewed as the transfer "within" the coalition, and does not affect the coalition's utility. In Section 6, we analyze the metagame that models coalition formation.

3.3 Game-Theoretic Properties

Cooperative Strategy Proofness. The first property we consider is cooperative strategy proofness (CSP-fairness), which was formulated and adopted in prior works [PS17, CGL+18, WAS22, CS23, SCW23, CRS24, CMST22b]. Conceptually, a protocol satisfies CSP-fairness if following the protocol maximizes the expected utility of a coalition of strategic players, assuming that the remaining players follows the protocol honestly. Consequently, deviation is not profitable, and the honest protocol achieves a coalition-resistant Nash Equilibrium. The formal definition of CSP-fairness is as follows.

Definition 3.1 (CSP fairness)**.** A protocol satisfies γ-CSP-fairness iff the following holds. Let $\mathcal{C}$ be any coalition that controls at most a $\gamma \in [0, 1)$ fraction of the mining power, and possibly includes either Alice or Bob. Then, for any probabilistic polynomial-time (PPT) strategy $S_{\mathcal{C}}$ of $\mathcal{C}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that except with $\mathsf{negl}(\lambda)$ probability, we have

$$\mathsf{util}^{\mathcal{C}}(S_{\mathcal{C}}, HS_{-\mathcal{C}}) \leq \mathsf{util}^{\mathcal{C}}(HS_{\mathcal{C}}, HS_{-\mathcal{C}}), \tag{1}$$

where $HS_{\mathcal{C}}$ denotes the honest strategy of $\mathcal{C}$, $HS_{-\mathcal{C}}$ denotes the honest strategy of anyone other than $\mathcal{C}$, and $\mathsf{util}^{\mathcal{C}}(X_{\mathcal{C}}, Y_{-\mathcal{C}})$ is the expected utility of the coalition $\mathcal{C}$ when $\mathcal{C}$ is executing strategy X and the remaining players (denoted by $-\mathcal{C}$) execute strategy Y.[1]

Dropout Resilience. In atomic swap, we want to guarantee dropout resilience, which protects an honest player when the counterparty drops out from the protocol. In practice, a dropout can happen due to mistakes, misconfiguration, or unforeseen circumstances, e.g., Alice may lose her hardware wallet. Conceptually, a protocol satisfies dropout resilience if an honest player's utility is always non-negative, even when the counterparty stops responding and drops out before the end of the protocol assuming at least $1/\mathsf{poly}(\lambda)$ fraction of the mining power is honest. The formal definition of dropout resilience is as follows.

Definition 3.2 (Dropout resilience)**.** A protocol is dropout resilient, iff as long as at least $1/\mathsf{poly}(\lambda)$ fraction of the mining power is honest, then with $1 - \mathsf{negl}(\lambda)$ probability, an honest Alice (or Bob) is guaranteed to have non-negative utility even when Bob (or Alice) is honest but drops out during the protocol's execution.

We emphasize that our dropout resilience notion is very strong: we want it to hold even when many miners (up to $1 - 1/\mathsf{poly}(\lambda)$ fraction) are not necessarily playing honestly. In such a scenario, transactions may take polynomially long to confirm, and players may time out during the protocol and try to back out.

3.3.1 New Definition: Bounded Maximin Fairness

While CSP-fairness assumes that players do not have incentives outside the present protocol, achieving resilience against such external incentives is very meaningful. In particular, such incentives may

[1] The formal definition of the utility function util is given in Section 3.4 in the context of atomic swap.

lead strategic players to adopt a broader class of strategies, including ones where players suffer loss within the protocol but are compensated by the external incentives. Given unbounded external incentives, strategic players can behave arbitrarily (like a "malicious" adversary using standard cryptographic terminology). The strongest game theoretic notion one can hope for is that honest players are not harmed even when others behave arbitrarily. This notion is called *maximin fairness* in some recent works [PS17, CGL+18, WAS22].

Maximin fairness, however, can be too stringent and challenging to satisfy. Thus, many real-world protocols that hope to incentivize honest behavior aim to provide incentive compatibility only when the parties' external incentives are bounded. Specifically for atomic swap, it is unclear whether a protocol that achieves maximin fairness given user-miner coalition exists at all. Hence, we define a relaxed notion of fairness called *bounded maximin fairness*. It protects honest individuals from rational players who may have *arbitrary* but *bounded* external incentives. Although we focus on applying this notion to atomic swap, it would be interesting apply it more broadly to formally analyze real-world protocols that currently try to provide *heuristic* guarantees in the face of bounded external incentives.

Defining bounded maximin fairness. Imagine a set of players $\mathcal{C}'$ who have external incentives that might entice them to deviate from honest behavior. We want to argue that even when their external incentives are *arbitrary*, then as long as the incentives are *bounded* and $\mathcal{C}'$ is *rational*, any group of players without external incentives should feel safe to participate honestly — as long as they participate honestly, their utility will not be negative.

When defining *rationality* of the externally incentivized coalition, interesting technicalities arise due to the fact that we model players and contracts as PPT interactive Turing machines. It is tempting to assume that the externally incentivized coalition $\mathcal{C}'$ uses the *optimal* strategy that maximizes its expected utility, and ensure no harm for any honest group or individual. However, this approach would make the possibly unrealistic assumption that $\mathcal{C}'$ *knows* the optimal strategy based on the external incentive function. Indeed, finding such a strategy may be computationally hard, since the external incentive function can take an arbitrary form.

Instead, we will protect honest participants against any PPT strategy of the coalition $\mathcal{C}'$ as long as the strategy is not *blatantly irrational*. A family of strategies $\bar{\mathcal{R}}$ is called blatantly irrational, if for any strategy $S \in \bar{\mathcal{R}}$, one can efficiently find another strategy S', such that S' does strictly better than S.[2] We can often show that some strategies are *blatantly irrational* without finding the optimal strategy. Later in our formal proofs, we show that a class of strategies is blatantly irrational, since simple modifications of such strategies lead to better outcomes for $\mathcal{C}'$. We then show that as long as $\mathcal{C}'$ does not adopt a blatantly irrational strategy, honest participants are protected.

We therefore devise the following definition which is parametrized by a strategy space $\mathcal{R}$ that contains all possible PPT strategies except for the set of blatantly irrational strategies $\overline{\mathcal{R}}$.

Definition 3.3 (Bounded maximin fairness)**.** A protocol satisfies α-bounded maximin fairness w.r.t. some strategy space $\mathcal{R}$, iff for any set of PPT players $\mathcal{C}$ without external incentives, and any externally incentivized PPT coalition $\mathcal{C}'$ that is disjoint from $\mathcal{C}$, controls at most α fraction of the mining power, and uses any strategy $S_{\mathcal{C}'} \in \mathcal{R}$, there is a negligible function $\mathsf{negl}(\cdot)$ such that except with $\mathsf{negl}(\lambda)$ probability[3], it holds that

$$\mathsf{util}^{\mathcal{C}}(HS_{\mathcal{C}}, S_{\mathcal{C}'}, HS_{\mathcal{D}}) \geq 0,$$

[2] How we construct blatantly irrational strategies is given in Section 5.2.1, and how one can find a better strategy efficiently is explained in Lemma 5.7 and Lemma 5.10.

[3] The negligible failure probability in our proofs arises due to the following bad events: 1) either the hash is inverted, or 2) honest miners have never mined a block even after polynomially many blocks have been mined.

where $\mathcal{D}$ denotes all players not in $\mathcal{C} \cup \mathcal{C}'$, $HS_{\mathcal{C}}$ and $HS_{\mathcal{D}}$ denote the honest strategy of $\mathcal{C}$ and $\mathcal{D}$, respectively, and $\mathsf{util}^{\mathcal{C}}(HS_{\mathcal{C}}, S_{\mathcal{C}'}, HS_{\mathcal{D}})$ denotes the utility of $\mathcal{C}$ given that the strategy $(HS_{\mathcal{C}}, S_{\mathcal{C}'}, HS_{\mathcal{D}})$ is adopted.

Regarding external incentives. As mentioned above, our modeling of external incentives is general and captures external incentives of any form. Precisely, any side contract where money is redistributed to players of the present protocol is considered as external incentive if **(1)** the contract is *pre-existing*, i.e., created before the start of the present protocol; or **(2)** the contract is created at any time, and an *outsider*, i.e., not a player of the present protocol, deposited money into it. As an example of the former, imagine that Alice was involved in some bet prior to the atomic swap protocol, and the bet depends on the state of a future block. Because the outcome of the atomic swap protocol may affect the blockchain state, Alice may be incentivized to manipulate the state even if doing so lead to a loss *within* the atomic swap protocol. As an example of the latter, imagine that Alice is Mallory's competitor, and Mallory is offering external incentives for anyone who can cause financial loss to Alice. Without loss of generality, we consider non-negative external incentives. Further, for pre-existing smart contracts, we do not distinguish between those funded by the participants in our protocol and those funded by external parties — gains from all pre-existing smart contracts are considered external.

3.4 Atomic Swap

An atomic swap allows two mutually distrustful parties to exchange coins across different blockchains without relying on a trusted third party. Consider two parties, Alice and Bob. Bob owns x_b coins on $\mathsf{BobChain}$ (denoted $\ddot{\mathrm{B}}x_b$), and Alice owns x_a coins on $\mathsf{AliceChain}$ (denoted $\ddot{\mathrm{A}}x_a$). The goal is for Bob to exchange his $\ddot{\mathrm{B}}x_b$ for Alice's $\ddot{\mathrm{A}}x_a$.

In this work, we consider three kinds of strategic players: 1) Alice-miner coalition (or Alice alone); 2) Bob-miner coalition (or Bob alone); and 3) miner-only coalition. Let $\$\mathsf{AV}(\cdot)$ denote the valuation function of Alice (or the Alice-miner coalition), defined as:

$$\$\mathsf{AV}(\ddot{\mathrm{B}}x_b + \ddot{\mathrm{A}}x_a) = \$v_a^{\mathsf{B}} \cdot x_b + \$v_a^{\mathsf{A}} \cdot x_a,$$

where $\$v_a^{\mathsf{B}} \geq 0$ and $\$v_a^{\mathsf{A}} \geq 0$ represent the value Alice places on each coin on $\mathsf{BobChain}$ and $\mathsf{AliceChain}$, respectively. Similarly, let $\$\mathsf{BV}(\cdot)$ denote the valuation function of Bob (or the Bob-miner coalition), and $\$\mathsf{MV}(\cdot)$ denote that of the miner-only coalition. We make the assumption: $\$\mathsf{AV}(\ddot{\mathrm{B}}x_b - \ddot{\mathrm{A}}x_a) > 0$, $\$\mathsf{BV}(\ddot{\mathrm{A}}x_a - \ddot{\mathrm{B}}x_b) > 0$, which justifies why Alice wants to exchange her $\ddot{\mathrm{A}}x_a$ with Bob's $\ddot{\mathrm{B}}x_b$, and vice versa.

Let $\mathcal{C}$ be any subset of players, with $S_{\mathcal{C}}$ and $S'_{-\mathcal{C}}$ denoting the strategies of $\mathcal{C}$ and $-\mathcal{C}$, respectively. Let $\ddot{\mathrm{A}}d_a^{\mathsf{A}}, \ddot{\mathrm{B}}d_a^{\mathsf{B}} \geq 0$ be the amounts deposited by Alice (or the Alice-miner coalition) into the respective smart contracts, and $\ddot{\mathrm{A}}r_a^{\mathsf{A}}, \ddot{\mathrm{B}}r_a^{\mathsf{B}} \geq 0$ be the amounts received during protocol execution. Let $\$e_a(\ldots) \geq 0$ represent the external incentives for Alice or the Alice-miner coalition, where the value may depend arbitrarily on the blockchain state. We define the utility $\mathsf{util}^{\mathcal{C}}(S_{\mathcal{C}}, S'_{-\mathcal{C}})$ when $\mathcal{C}$ is Alice or Alice-miner coalition as follows:

$$\mathsf{util}^{\mathcal{C}}(S_{\mathcal{C}}, S'_{-\mathcal{C}}) = \$\mathsf{AV}(\ddot{\mathrm{A}}r_a^{\mathsf{A}} - \ddot{\mathrm{A}}d_a^{\mathsf{A}} + \ddot{\mathrm{B}}r_a^{\mathsf{B}} - \ddot{\mathrm{B}}d_a^{\mathsf{B}}) + \$e_a(\ldots).$$

We define $\ddot{\mathrm{A}}d_b^{\mathsf{A}}, \ddot{\mathrm{B}}d_b^{\mathsf{B}}, \ddot{\mathrm{A}}r_b^{\mathsf{A}}, \ddot{\mathrm{B}}r_b^{\mathsf{B}}, \$e_b(\ldots) \geq 0$ analogously for Bob (or the Bob-miner coalition), and $\ddot{\mathrm{A}}r_m^{\mathsf{A}}, \ddot{\mathrm{B}}r_m^{\mathsf{B}}, \ddot{\mathrm{A}}d_m^{\mathsf{A}}, \ddot{\mathrm{B}}d_m^{\mathsf{B}}, \$e_m(\ldots) \geq 0$ for the miner-only coalition. The utility $\mathsf{util}^{\mathcal{C}}(S_{\mathcal{C}}, S'_{-\mathcal{C}})$ when $\mathcal{C}$ consists of Bob or a Bob-miner coalition is defined as

$$\mathsf{util}^{\mathcal{C}}(S_{\mathcal{C}}, S'_{-\mathcal{C}}) = \$\mathsf{BV}(\ddot{\mathrm{A}}r_b^{\mathsf{A}} - \ddot{\mathrm{A}}d_b^{\mathsf{A}} + \ddot{\mathrm{B}}r_b^{\mathsf{B}} - \ddot{\mathrm{B}}d_b^{\mathsf{B}}) + \$e_b(\ldots),$$

and the utility $\mathsf{util}^{\mathcal{C}}(S_{\mathcal{C}}, S'_{-\mathcal{C}})$ when $\mathcal{C}$ is a miner-only coalition is defined as

$$\mathsf{util}^{\mathcal{C}}(S_{\mathcal{C}}, S'_{-\mathcal{C}}) = \$\mathsf{MV}(\ddot{\mathbb{A}}r_m^{\mathsf{A}} - \ddot{\mathbb{A}}d_m^{\mathsf{A}} + \ddot{\mathbb{B}}r_m^{\mathsf{B}} - \ddot{\mathbb{B}}d_m^{\mathsf{B}}) + \$e_m(\ldots).$$

4 Secure Atomic Swap

We now present the smart contracts and the protocol of our atomic swap. Say Alice owns x_a coins on AliceChain (denoted $\ddot{\mathbb{A}}x_a$), and Bob owns x_b coins on BobChain (denoted $\ddot{\mathbb{B}}x_b$). Bob wants to exchange his $\ddot{\mathbb{B}}x_b$ for Alice's $\ddot{\mathbb{A}}x_a$. Either Alice or Bob can be a strategic player, and collude with the miners on both chains. $\$E$ is the upper bound of the external incentive of the coalition, and $\alpha \in [0, 1)$ is the fraction of the mining power controlled by the coalition.

To initiate an atomic swap, Alice samples $pre_s \leftarrow \{0,1\}^\lambda, pre_a \leftarrow \{0,1\}^\lambda$, and Bob samples $pre_b \leftarrow \{0,1\}^\lambda$ and $pre_c \leftarrow \{0,1\}^\lambda$, where λ is the security parameter. Alice and Bob choose the parameters that respect the constraints in Figure 2. Then, Alice and Bob deploy two contracts: $\textsc{Contract}^{\mathrm{A}}$ and $\textsc{Contract}^{\mathrm{B}}$ specified in Figure 3. We rename the block height where $\textsc{Contract}^{\mathrm{A}}$ is deployed on AliceChain to be AliceChain time 0 (only deployed, Alice and Bob have not deposited their coins yet). Similarly, BobChain time 0 is the block height where $\textsc{Contract}^{\mathrm{B}}$ is deployed on BobChain.

We give the protocols specifying the honest behaviors of Alice, Bob, and miners in Figure 4, Figure 5, and Figure 6, respectively. This atomic swap protocol satisfies CSP-fairness, bounded maximin fairness, and dropout resilience:

Theorem 4.1. *Suppose that the hash function $H(\cdot)$ is a one-way function, and the choice of the parameters satisfy the constraints in Figure 2. Then, the following statements hold:*

- *For any $\gamma \in [0, 1 - 1/\mathsf{poly}(\lambda)]$, if the parameters further satisfy $\gamma^{\tau^{\mathsf{A}}} \leq \frac{\ddot{\mathbb{A}}c_a^{\mathsf{A}}}{\ddot{\mathbb{A}}c_a^{\mathsf{A}} + \ddot{\mathbb{A}}x_a}$ and $\gamma^{\tau^{\mathsf{B}}} \leq \frac{\ddot{\mathbb{B}}c_b^{\mathsf{B}}}{\ddot{\mathbb{B}}c_b^{\mathsf{B}} + \ddot{\mathbb{B}}x_b}$, then atomic swap protocol satisfies γ-CSP-fairness.*
- *If $\alpha \in [0, 1 - 1/\mathsf{poly}(\lambda)]$, then atomic swap protocol satisfies α-bounded maximin fairness against external incentives.*
- *If all players are PPT, then atomic swap protocol is dropout resilient.*

The formal proof of Theorem 4.1 is given in Section 5. In Section 6, we analyze the metagame that captures the formation of the coalition. We show that our atomic swap protocol disincentivizes 100% of the miners to collude with the strategic players, and thus justifies the assumption of having $1/\mathsf{poly}(\lambda)$ fraction of honest miners.

In addition to the game-theoretic properties above, we also show that honest players can get their collateral back in a timely manner as specified in the following theorem.

Theorem 4.2. *Suppose Alice, Bob, and all miners are honest. Alice gets $\ddot{\mathbb{B}}x_b$, Bob gets $\ddot{\mathbb{A}}x_a$, and both parties get their collateral back in 3* BobChain *time plus 2* AliceChain *time.*

Proof. If Alice and Bob are both honest, they will send the deposit transactions to $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ at BobChain time $t = 0$. Because all the miners are honest, $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ both enter the execution phase no later than one block on BobChain or one block on AliceChain is mined. As soon as $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ both enter the execution phase, Bob sends ping to $B^{\mathsf{B}}_{\mathsf{defuse}}$. Because all the miners are honest, $B^{\mathsf{B}}_{\mathsf{defuse}}$ will be activated no later than one block on BobChain is mined. As soon as $B^{\mathsf{B}}_{\mathsf{defuse}}$ is activated, Bob sends pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$.

Constraints for Contract$^{\mathbf{B}}$ (on BobChain):

- $h_s = H(pre_s)$, $h_b = H(pre_b)$ and $h_c = H(pre_c)$.
- $T_1^{\mathsf{B}} > T_0^{\mathsf{B}} > T^{\mathsf{B}} > 0$.
- $\ddot{\mathrm{B}}\epsilon^{\mathsf{B}} > \ddot{\mathrm{B}}0$, $\ddot{\mathrm{B}}c_a^{\mathsf{B}} > \ddot{\mathrm{B}}\epsilon^{\mathsf{B}}$, and $\ddot{\mathrm{B}}c_b^{\mathsf{B}} > \ddot{\mathrm{B}}\epsilon^{\mathsf{B}}$.
- $\$\mathsf{AV}(\ddot{\mathrm{B}}c_a^{\mathsf{B}}) > \frac{\$\mathsf{AV}(\ddot{\mathrm{A}}x_a + \alpha\ddot{\mathrm{B}}x_b) + \$E}{1-\alpha}$ and
  $\$\mathsf{BV}(\ddot{\mathrm{B}}c_b^{\mathsf{B}}) > \frac{\$\mathsf{BV}(\ddot{\mathrm{A}}x_a + \alpha\ddot{\mathrm{B}}x_b) + \$E}{1-\alpha}$

Constraints for Contract$^{\mathbf{A}}$ (on AliceChain):

- $h_s = H(pre_s)$ and $h_a = H(pre_a)$.
- $T_1^{\mathsf{A}} > T_0^{\mathsf{A}} > 0$.
- AliceChain time T_1^{A} > BobChain time T_1^{B}, i.e., the AliceChain block of length T_1^{A} is mined after the BobChain block of length T_1^{B}.[a]
- $\ddot{\mathrm{A}}\epsilon^{\mathsf{A}} > \ddot{\mathrm{A}}0$, $\ddot{\mathrm{A}}c_a^{\mathsf{A}} > \ddot{\mathrm{A}}\epsilon^{\mathsf{A}}$, and $\ddot{\mathrm{A}}c_b^{\mathsf{A}} > \ddot{\mathrm{A}}\epsilon^{\mathsf{A}}$.
- $\$\mathsf{AV}(\ddot{\mathrm{A}}c_a^{\mathsf{A}}) > \frac{\$\mathsf{AV}(\ddot{\mathrm{B}}x_b + \alpha\ddot{\mathrm{A}}x_a) + \$E}{1-\alpha}$ and
  $\$\mathsf{BV}(\ddot{\mathrm{A}}c_b^{\mathsf{A}}) > \frac{\$\mathsf{BV}(\ddot{\mathrm{B}}x_b + \alpha\ddot{\mathrm{A}}x_a) + \$E}{1-\alpha}$.

Choice of timeouts:

- $\tau^{\mathsf{B}} \geq 1$, $\tau^{\mathsf{A}} \geq 1$.

[a] In practice, this constraint should be respected except with negligible probability despite the variance in inter-block times.

Figure 2: Parameter constraints for atomic swap. $H(\cdot)$ is a cryptographic hash function. All times are expressed in the time of the respective chain.

Because we assume the network delay is zero, Alice and Bob both enter the execution phase when Bob sends pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$.

When Alice enters the execution phase, she sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ immediately. Because all the miners are honest, $P^{\mathsf{B}}_{\mathsf{default}}$ will be activated no later than one block on BobChain is mined. As soon as $P^{\mathsf{B}}_{\mathsf{default}}$ is activated, Alice sends ping to $P^{\mathsf{A}}_{\mathsf{default}}$. Again, ecause all the miners are honest, $P^{\mathsf{A}}_{\mathsf{default}}$ will be activated no later than one block on AliceChain is mined. When $P^{\mathsf{B}}_{\mathsf{default}}$ and $P^{\mathsf{A}}_{\mathsf{default}}$ are activated, Alice gets $\ddot{\mathrm{B}}x_b$, Bob gets $\ddot{\mathrm{A}}x_a$, and both parties get their collateral back. □

Contract$^{\mathbf{B}}$ (on BobChain)

/* parametrized with $(h_s, h_b, h_c, T_1^{\mathsf{B}}, \tau^{\mathsf{B}}, \ddot{\mathrm{B}}x_b, \ddot{\mathrm{B}}c_b^{\mathsf{B}}, \ddot{\mathrm{B}}\epsilon^{\mathsf{B}})$*/

Deposits: Bob deposits $\ddot{\mathrm{B}}x_b + \ddot{\mathrm{B}}c_b^{\mathsf{B}}$, and Alice deposits $\ddot{\mathrm{B}}c_a^{\mathsf{B}}$. Once both parties have deposited the required amount, the contract becomes active.

$B_{\mathsf{defuse}}^{\mathsf{B}}$: On receiving ping from anyone, do nothing. ▷ $B_{\mathsf{defuse}}^{\mathsf{B}}$ is to invalidate $B_{\mathsf{burn}}^{\mathsf{B}}$.

$B_{\mathsf{burn}}^{\mathsf{B}}$: On receiving z from anyone P such that $H(z) = h_c$, send $\ddot{\mathrm{B}}\epsilon^{\mathsf{B}}$ to player P. All remaining coins are burnt.

$P_{\mathsf{default}}^{\mathsf{B}}$: On receive z_1 from Alice such that $H(z_1) = h_s$ and z_2 from Bob such that $H(z_2) = h_c$, send $\ddot{\mathrm{B}}x_b + \ddot{\mathrm{B}}c_a^{\mathsf{B}}$ to Alice and $\ddot{\mathrm{B}}c_b^{\mathsf{B}}$ to Bob.

$P_{\mathsf{refund}}^{\mathsf{B}}$: Time T_1^{B} or greater: On receive z from Bob such that $H(z) = h_b$ or on receiving ping from Alice, do nothing.

$C_{\mathsf{refund}}^{\mathsf{B}}$: At least τ^{B} after $P_{\mathsf{refund}}^{\mathsf{B}}$ is activated: on receiving ping from anyone, send $\ddot{\mathrm{B}}x_b + \ddot{\mathrm{B}}c_b^{\mathsf{B}}$ to Bob and $\ddot{\mathrm{B}}c_a^{\mathsf{B}}$ to Alice.

$C_{\mathsf{burn}}^{\mathsf{B}}$: On receive (z_1, z_2, z_3) from anyone P such that $H(z_1) = h_s$, $H(z_2) = h_b$, and $H(z_3) = h_c$ send $\ddot{\mathrm{B}}\epsilon^{\mathsf{B}}$ to player P. All remaining coins are burnt.

Contract$^{\mathbf{A}}$ (on AliceChain)

/* parametrized with $(h_s, h_a, T_1^{\mathsf{A}}, \tau^{\mathsf{A}}, \ddot{\mathrm{A}}x_a, \ddot{\mathrm{A}}c_a^{\mathsf{A}}, \$\ddot{\mathrm{A}}\epsilon^{\mathsf{A}})$*/

Deposits: Alice deposits $\ddot{\mathrm{A}}x_a + \ddot{\mathrm{A}}c_a^{\mathsf{A}}$, and Bob deposits $\ddot{\mathrm{A}}c_b^{\mathsf{A}}$. Once both parties have deposited the required amount, the contract becomes active.

$A_{\mathsf{defuse}}^{\mathsf{A}}$: Time T_1^{A} or greater: on receiving ping from Alice or Bob, do nothing. ▷ $A_{\mathsf{defuse}}^{\mathsf{A}}$ is to invalidate $A_{\mathsf{burn}}^{\mathsf{A}}$.

$A_{\mathsf{burn}}^{\mathsf{A}}$: On receiving z_1 from anyone P such that $H(z_1) = h_a$, or on receiving z_2 from anyone P such that $H(z_2) = h_b$, send $\ddot{\mathrm{A}}\epsilon^{\mathsf{A}}$ to player P. All remaining coins are burnt.

$P_{\mathsf{default}}^{\mathsf{A}}$: On receiving z from Bob such that $H(z) = h_s$ or on receiving ping from Alice, send $\ddot{\mathrm{A}}x_a + \ddot{\mathrm{A}}c_b^{\mathsf{A}}$ to Bob and send $\ddot{\mathrm{A}}c_a^{\mathsf{A}}$ to Alice.

$P_{\mathsf{refund}}^{\mathsf{A}}$: Time T_1^{A} or greater: on receiving z from Alice such that $H(z) = h_a$ or on receiving ping from Bob, do nothing.

$C_{\mathsf{refund}}^{\mathsf{A}}$: At least τ^{A} after $P_{\mathsf{refund}}^{\mathsf{A}}$ is activated: on receiving ping from anyone, send $\ddot{\mathrm{A}}x_a + \ddot{\mathrm{A}}c_a^{\mathsf{A}}$ to Alice and $\ddot{\mathrm{A}}c_b^{\mathsf{A}}$ to Bob.

$C_{\mathsf{burn}}^{\mathsf{A}}$: On receiving (z_1, z_2) from anyone P such that $(z_1) = h_s$ and $H(z_2) = h_a$, or on receiving (z_2, z_3) from anyone P such that $H(z_2) = h_a$ and $H(z_3) = h_b$, send $\ddot{\mathrm{A}}\epsilon^{\mathsf{A}}$ to player P. All remaining coins are burnt.

Figure 3: Smart contracts for atomic swap. Activation points of the same type are mutually exclusive. Activation points can be triggered only after both parties have deposited.

Atomic Swap Protocol — Alice

Preparation Phase:

1. When both $\textsc{Contract}^{\mathrm{A}}$ and $\textsc{Contract}^{\mathrm{B}}$ have been deployed on the respective chains, Alice sends the deposit $\ddot{\mathrm{A}}x_a + \ddot{\mathrm{A}}c_a^{\mathsf{A}}$ to $\textsc{Contract}^{\mathrm{A}}$; and the collateral $\ddot{\mathrm{B}}c_a^{\mathsf{B}}$ to $\textsc{Contract}^{\mathrm{B}}$.
2. Wait until one of the following happens:
   - Either $\textsc{Contract}^{\mathrm{B}}$ or $\textsc{Contract}^{\mathrm{A}}$ has not been active, and it is at least $\mathsf{BobChain}$ time T^{B}: Alice enters the abort phase.
   - Bob has not sent pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$, and it is at least $\mathsf{BobChain}$ time T_0^{B}: Alice enters the abort phase.
   - Bob sent pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$ and it is before $\mathsf{BobChain}$ time T_0^{B}: Alice enters the execution phase.

Execution phase:

1. Alice sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$. As soon as $P^{\mathsf{B}}_{\mathsf{default}}$ has been activated, Alice sends ping to $P^{\mathsf{A}}_{\mathsf{default}}$.
2. If τ^{B} $\mathsf{BobChain}$ *time* has passed since $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Alice sends ping to $C^{\mathsf{B}}_{\mathsf{refund}}$. (Note that as soon as $C^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Bob sends ping to $P^{\mathsf{A}}_{\mathsf{refund}}$.)
3. If τ^{A} $\mathsf{AliceChain}$ *time* has passed since activating $P^{\mathsf{A}}_{\mathsf{refund}}$, Alice sends ping to $C^{\mathsf{A}}_{\mathsf{refund}}$.

Abort Phase:

1. If $\textsc{Contract}^{\mathrm{B}}$ ($\textsc{Contract}^{\mathrm{A}}$, resp.) has not been active, Alice withdraws her deposit from $\textsc{Contract}^{\mathrm{B}}$ ($\textsc{Contract}^{\mathrm{A}}$, resp.).
2. At $\mathsf{AliceChain}$ time T_0^{A}, Alice sends ping to $P^{\mathsf{B}}_{\mathsf{refund}}$ and and ping to $A^{\mathsf{A}}_{\mathsf{defuse}}$.
3. If Bob has not sent ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ by $\mathsf{AliceChain}$ time T_1^{A}, Alice waits until $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated and sends pre_a to $P^{\mathsf{A}}_{\mathsf{refund}}$.
4. If τ^{A} $\mathsf{AliceChain}$ *time* has passed since $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated, Alice sends ping to $C^{\mathsf{A}}_{\mathsf{refund}}$; similarly, if τ^{B} $\mathsf{BobChain}$ *time* has passed since $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Alice sends ping to $C^{\mathsf{B}}_{\mathsf{refund}}$.

Ignore all other events.

Figure 4: Atomic swap protocol for Alice.

Atomic Swap Protocol — Bob

Preparation Phase:

1. When both $\text{CONTRACT}^{\text{A}}$ and $\text{CONTRACT}^{\text{B}}$ have been deployed on the respective chains, Bob sends the deposit transaction of $\ddot{\text{B}}x_b + \ddot{\text{B}}c_b^{\mathsf{B}}$ to $\text{CONTRACT}^{\text{B}}$ and sends the collateral transaction of $\ddot{\text{A}}c_b^{\mathsf{A}}$ to $\text{CONTRACT}^{\text{A}}$.
2. Wait until one of the following happens:
   - Both $\text{CONTRACT}^{\text{B}}$ and $\text{CONTRACT}^{\text{A}}$ are active: Bob sends ping to $B^{\mathsf{B}}_{\mathsf{defuse}}$. As soon as $B^{\mathsf{B}}_{\mathsf{defuse}}$ is activated, Bob sends pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$ and enters the execution phase.
   - Either $\text{CONTRACT}^{\text{B}}$ or $\text{CONTRACT}^{\text{A}}$ has not been active, and it is at least BobChain time T^{B}: Bob enters the abort phase;

Execution phase:

1. Wait until one of the following happens:
   - Alice sent pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$: Bob sends pre_s to $P^{\mathsf{A}}_{\mathsf{default}}$.
   - Alice has not sent pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$, and it is at least BobChain time T_1^{B}: Bob sends ping to $A^{\mathsf{A}}_{\mathsf{defuse}}$. As soon as $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated, Bob sends pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$.
2. If τ^{B} BobChain *time* has passed since $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Bob sends ping to $C^{\mathsf{B}}_{\mathsf{refund}}$. As soon as $C^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Bob sends ping to $P^{\mathsf{A}}_{\mathsf{refund}}$.
3. If τ^{A} AliceChain *time* has passed since $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated, Bob sends ping to $C^{\mathsf{A}}_{\mathsf{refund}}$.

Abort Phase:

1. If $\text{CONTRACT}^{\text{B}}$ ($\text{CONTRACT}^{\text{A}}$, resp.) has not been active, Bob withdraws his deposit from $\text{CONTRACT}^{\text{B}}$ ($\text{CONTRACT}^{\text{A}}$, resp.).
2. At AliceChain time T_0^{A}, Bob sends ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ and ping to $A^{\mathsf{A}}_{\mathsf{defuse}}$.
3. If Alice has not sent ping to $P^{\mathsf{B}}_{\mathsf{refund}}$ by AliceChain time T_1^{A}, Bob waits until $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated and sends pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$.
4. If τ^{A} AliceChain *time* has passed since $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated, Bob sends ping to $C^{\mathsf{A}}_{\mathsf{refund}}$; similarly, if τ^{B} BobChain *time* has passed since $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Bob sends ping to $C^{\mathsf{B}}_{\mathsf{refund}}$.

Ignore all other events.

Figure 5: Atomic swap protocol for Bob.

Atomic Swap Protocol — Miner

- The miner watches all transactions posted to $B^{\mathsf{B}}_{\mathsf{defuse}}$, $B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{default}}$, $P^{\mathsf{B}}_{\mathsf{refund}}$, $C^{\mathsf{B}}_{\mathsf{refund}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $A^{\mathsf{A}}_{\mathsf{defuse}}$, $A^{\mathsf{A}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{default}}$, $P^{\mathsf{A}}_{\mathsf{refund}}$, $C^{\mathsf{A}}_{\mathsf{refund}}$, and $C^{\mathsf{A}}_{\mathsf{burn}}$ (i.e., all the activation points for both contracts), to see if they contain a valid pre_s, pre_a, pre_b, and pre_c.
- If $C^{\mathsf{B}}_{\mathsf{refund}}$ has not been activated, as soon as the miner has observed pre_s, pre_b and pre_c, it posts (pre_s, pre_b, pre_c) to $C^{\mathsf{B}}_{\mathsf{burn}}$. Similarly, if $C^{\mathsf{A}}_{\mathsf{refund}}$ has not been activated, as soon as the miner has observed both pre_a and pre_s, it posts (pre_a, pre_s) to $C^{\mathsf{A}}_{\mathsf{burn}}$; as soon as the miner has observed pre_a and pre_b, it posts (pre_a, pre_b) to $C^{\mathsf{A}}_{\mathsf{burn}}$.
- If $B^{\mathsf{B}}_{\mathsf{defuse}}$ has not been activated, as soon as the miner observes pre_c, it posts pre_c to $B^{\mathsf{B}}_{\mathsf{burn}}$. If $A^{\mathsf{A}}_{\mathsf{defuse}}$ has not been activated, as soon as miner observes pre_a or pre_b, it posts pre_a or pre_b to $A^{\mathsf{A}}_{\mathsf{burn}}$.
- Whenever the miner mines a block, it always includes its own transactions ahead of others.

Figure 6: Atomic swap protocol for miners.

5 Proof of Theorem 4.1

5.1 Achieving CSP Fairness

This subsection is dedicated to proving CSP-fairness which formally stated as follows.

Theorem 5.1 (CSP fairness)**.** *Suppose that the hash function $H(\cdot)$ is a one-way function. Suppose the choice of the parameters satisfy the constraints in Figure 2, and further satisfy $\gamma^{\tau^{\mathsf{A}}} \leq \frac{\ddot{\mathtt{A}} c_a^{\mathsf{A}}}{\ddot{\mathtt{A}} c_a^{\mathsf{A}} + \ddot{\mathtt{A}} x_a}$ and $\gamma^{\tau^{\mathsf{B}}} \leq \frac{\ddot{\mathtt{B}} c_b^{\mathsf{B}}}{\ddot{\mathtt{B}} c_b^{\mathsf{B}} + \ddot{\mathtt{B}} x_b}$. Then, the atomic swap protocol satisfies γ-CSP-fairness.*

Before proving Theorem 5.1, we define the following events.

- $\mathsf{Normal}^{\mathsf{B}}$: $P^{\mathsf{B}}_{\mathsf{default}}$ is activated.
- $\mathsf{Refund}^{\mathsf{B}}$: either $(P^{\mathsf{B}}_{\mathsf{refund}} + C^{\mathsf{B}}_{\mathsf{refund}})$ are activated, or one of Alice and Bob withdraws their deposits from $\textsc{Contract}^{\mathrm{B}}$ successfully before $\textsc{Contract}^{\mathrm{B}}$ becomes active.
- $\mathsf{Burn}^{\mathsf{B}}$: $B^{\mathsf{B}}_{\mathsf{burn}}$ or $C^{\mathsf{B}}_{\mathsf{burn}}$ is activated.
- $\mathsf{Normal}^{\mathsf{A}}$: $P^{\mathsf{A}}_{\mathsf{default}}$ is activated.
- $\mathsf{Refund}^{\mathsf{A}}$: either $(P^{\mathsf{A}}_{\mathsf{refund}} + C^{\mathsf{A}}_{\mathsf{refund}})$ are activated, or one of Alice and Bob withdraws their deposits from $\textsc{Contract}^{\mathrm{A}}$ successfully before $\textsc{Contract}^{\mathrm{A}}$ becomes active.
- $\mathsf{Burn}^{\mathsf{A}}$: $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated.

Normally, when Alice and Bob follow the protocol, $P^{\mathsf{B}}_{\mathsf{default}}$ and $P^{\mathsf{A}}_{\mathsf{default}}$ will be activated, and they exchange the coins successfully. However, if one of the parties drops out, the other party will trigger $(P^{\mathsf{B}}_{\mathsf{refund}} + C^{\mathsf{B}}_{\mathsf{refund}})$ and $(P^{\mathsf{A}}_{\mathsf{refund}} + C^{\mathsf{A}}_{\mathsf{refund}})$ to get refunded. Finally, $B^{\mathsf{B}}_{\mathsf{burn}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $A^{\mathsf{A}}_{\mathsf{burn}}$ and $C^{\mathsf{A}}_{\mathsf{burn}}$ are the bombs, and both Alice and Bob lose their collateral when a bomb is triggered.

Lemma 5.2. *Suppose the parameters are set according to Figure 2. Then, the following statements hold.*

- *Suppose the coalition $\mathcal{A}$ consists of Alice and an arbitrary $\gamma \in [0,1]$ fraction of the mining power. The utility of $\mathcal{A}$ can be more than the honest case, that is, $\$\mathsf{AV}(\ddot{\mathtt{B}} x_b - \ddot{\mathtt{A}} x_a)$, only if $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{A}}$ both happen.*
- *Suppose the coalition $\mathcal{B}$ consists of Bob and an arbitrary $\gamma \in [0,1]$ fraction of the mining power. The utility of $\mathcal{B}$ can be more than the honest case, that is, $\$\mathsf{BV}(\ddot{\mathtt{A}} x_a - \ddot{\mathtt{B}} x_b)$, only if $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ both happen.*

Proof. Notice that if any of $\mathsf{Normal}^{\mathsf{B}}$, $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Burn}^{\mathsf{B}}$ happens, no coin is left in $\mathsf{BobChain}$, so no one can get more coin from $\mathsf{BobChain}$ anymore. Thus, consider all possible cases, including none of $\mathsf{Normal}^{\mathsf{B}}$, $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Burn}^{\mathsf{B}}$ happens, we have the following table.

which is activated	net profit of Alice's coalition	net profit of Bob's coalition
none	$-\ddot{\mathtt{B}} c_a^{\mathsf{B}}$	$-\ddot{\mathtt{B}} x_b - \ddot{\mathtt{B}} c_b^{\mathsf{B}}$
$\mathsf{Normal}^{\mathsf{B}}$	$\ddot{\mathtt{B}} x_b$	$-\ddot{\mathtt{B}} x_b$
$\mathsf{Refund}^{\mathsf{B}}$	0	0
$\mathsf{Burn}^{\mathsf{B}}$	$\leq \ddot{\mathtt{B}} \epsilon^{\mathsf{B}} - \ddot{\mathtt{B}} c_a^{\mathsf{B}}$	$\leq \ddot{\mathtt{B}} \epsilon^{\mathsf{B}} - \ddot{\mathtt{B}} x_b - \ddot{\mathtt{B}} c_b^{\mathsf{B}}$

Table 1: The net profit of Bob's coalition from $\mathsf{BobChain}$.

Similarly, if any of $\mathsf{Normal}^{\mathsf{A}}$, $\mathsf{Refund}^{\mathsf{A}}$ and $\mathsf{Burn}^{\mathsf{A}}$ happens, no coin is left in AliceChain, so no one can get more coin from AliceChain anymore. Thus, consider all possible cases, including none of $\mathsf{Normal}^{\mathsf{A}}$, $\mathsf{Refund}^{\mathsf{A}}$ and $\mathsf{Burn}^{\mathsf{A}}$ happens, we have the following table.

which is activated	net profit of Alice's coalition	net profit of Bob's coalition
none	$-\ddot{\mathbb{A}}x_a - \ddot{\mathbb{A}}c_a^{\mathsf{A}}$	$-\ddot{\mathbb{A}}c_b^{\mathsf{A}}$
$\mathsf{Normal}^{\mathsf{A}}$	$-\ddot{\mathbb{A}}x_a$	$\ddot{\mathbb{A}}x_a$
$\mathsf{Refund}^{\mathsf{A}}$	0	0
$\mathsf{Burn}^{\mathsf{A}}$	$\leq \ddot{\mathbb{A}}\epsilon^{\mathsf{A}} - \ddot{\mathbb{A}}x_a - \ddot{\mathbb{A}}c_a^{\mathsf{A}}$	$\leq \ddot{\mathbb{A}}\epsilon^{\mathsf{A}} - \ddot{\mathbb{A}}c_b^{\mathsf{A}}$

Table 2: The net profit of Alice's coalition and Bob's coalition from AliceChain.

Alice-miner coalition. Suppose the coalition $\mathcal{A}$ consists of Alice and an arbitrary $\gamma \in [0,1]$ fraction of the mining power. If $\mathcal{A}$ follows the protocol, $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ will happen, and the utility of $\mathcal{C}$ is $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a) > 0$. When $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{A}}$ both happen, $\mathcal{A}$'s utility is $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b)$. Now, we will show that this is the only possible scenario for $\mathcal{A}$'s utility to exceed the honest case. In other words, if either $\mathsf{Normal}^{\mathsf{B}}$ or $\mathsf{Refund}^{\mathsf{A}}$ does not happen, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a)$. There are two cases.

- **Case 1:** $\mathsf{Normal}^{\mathsf{B}}$ **does not happen.** Because $\ddot{\mathbb{B}}c_a^{\mathsf{B}} > \ddot{\mathbb{B}}\epsilon^{\mathsf{B}}$, the net profit from BobChain is at most 0 if $\mathsf{Normal}^{\mathsf{B}}$ does not happen. Then, because $\ddot{\mathbb{A}}x_a > \ddot{\mathbb{A}}\epsilon^{\mathsf{A}}$, we have $\ddot{\mathbb{A}}\epsilon^{\mathsf{A}} - \ddot{\mathbb{A}}x_a - \ddot{\mathbb{A}}c_a^{\mathsf{A}} < 0$. Thus, the net profit from AliceChain is also at most 0. Consequently, the utility of $\mathcal{C}$ is at most zero, which is less than $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a)$.
- **Case 2:** $\mathsf{Refund}^{\mathsf{A}}$ **does not happen.** Because $\ddot{\mathbb{A}}c_a^{\mathsf{A}} > \ddot{\mathbb{A}}\epsilon^{\mathsf{A}}$, we have $\ddot{\mathbb{A}}\epsilon^{\mathsf{A}} - \ddot{\mathbb{A}}x_a - \ddot{\mathbb{A}}c_a^{\mathsf{A}} < -\ddot{\mathbb{A}}x_a$. Thus, assuming $\mathsf{Refund}^{\mathsf{A}}$ does not happen, the net profit from AliceChain is at most $-\ddot{\mathbb{A}}x_a$. However, the net profit from BobChain is at most $\ddot{\mathbb{B}}x_b$. Thus, the utility of $\mathcal{C}$ is at most $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a)$, which is the same as the honest case.

Bob-miner coalition. Using a completely symmetric proof, we can show that the only way for a Bob-miner coalition's utility to exceed the honest case is when $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ both happen. □

Lemma 5.3 (Alice-miner coalition)**.** *Suppose that the hash function $H(\cdot)$ is a one-way function. Let $\mathcal{A}$ be any coalition that consists of Alice and γ fraction of mining power. Then, as long as $\gamma^{\tau^{\mathsf{A}}} \leq \frac{\ddot{\mathbb{A}}c_a^{\mathsf{A}}}{\ddot{\mathbb{A}}c_a^{\mathsf{A}} + \ddot{\mathbb{A}}x_a}$, for any PPT coalition strategy $S_{\mathcal{A}}$, except with negligible probability, it must be*

$$\mathsf{util}^{\mathcal{A}}(S_{\mathcal{A}}, HS_{-\mathcal{A}}) \leq \mathsf{util}^{\mathcal{A}}(HS_{\mathcal{A}}, HS_{-\mathcal{A}}),$$

where $HS_{-\mathcal{A}}$ denotes the honest strategy for everyone not in $\mathcal{A}$.

Proof. Recall that the utility of $\mathcal{A}$ is $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a) > 0$ under an honest execution. Now, suppose $\mathcal{A}$ may deviate from the protocol. We may assume that the coalition does not post any new smart contract on the fly and deposit money into it — if it did so, it cannot recover more than its deposit since any player not in $\mathcal{A}$ will not invoke the smart contract. We analyze the possible cases depending on which phase Bob enters.

Bob enters the abort phase. Because $\ddot{\mathbb{A}}x_a > \ddot{\mathbb{A}}\epsilon^{\mathsf{A}}$, the net profit of $\mathcal{A}$ from AliceChain is at most zero, no matter whether $\textsc{Contract}^{\mathsf{A}}$ is active or not. We will show that the net profit of

$\mathcal{A}$ from BobChain is also at most zero except with negligible probability. If $\text{CONTRACT}^{\text{B}}$ never becomes active, the net profit of $\mathcal{A}$ from BobChain is at most zero. Now, assume $\text{CONTRACT}^{\text{B}}$ becomes active. When Bob enters the abort phase, he never sends any transaction containing pre_c. Ignoring the negligible probability that $\mathcal{A}$ finds pre_c by itself, $B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{default}}$, and $C^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. Because Alice does not get any coin from $B^{\mathsf{B}}_{\mathsf{defuse}}$, $P^{\mathsf{B}}_{\mathsf{refund}}$ or $C^{\mathsf{B}}_{\mathsf{refund}}$, the net profit of $\mathcal{A}$ from BobChain is at most zero.

To sum up, except with negligible probability, the utility of $\mathcal{A}$ is at most zero, which is less than the honest case.

Bob enters the execution phase. If Bob enters the execution phase, both $\text{CONTRACT}^{\text{B}}$ and $\text{CONTRACT}^{\text{A}}$ must be active. By Lemma 5.2, the utility of $\mathcal{A}$ can exceed the honest case only when $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{A}}$ both happen, so we assume it is the case. Because $\text{CONTRACT}^{\text{A}}$ is active, for $\mathsf{Refund}^{\mathsf{A}}$ to happen, $P^{\mathsf{A}}_{\mathsf{refund}}$ must be activated. When Bob enters the execution phase, $P^{\mathsf{A}}_{\mathsf{refund}}$ can be activated only either 1) by Bob sending ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ after $C^{\mathsf{B}}_{\mathsf{refund}}$ has been activated, or 2) by Alice sending pre_a to $P^{\mathsf{A}}_{\mathsf{refund}}$. Consider the first scenario. Since $C^{\mathsf{B}}_{\mathsf{refund}}$ has been activated, Alice cannot get any money from $\text{CONTRACT}^{\text{B}}$. However, from $\text{CONTRACT}^{\text{A}}$, Alice can get at most zero. Thus, the utility of $\mathcal{A}$ is less than the honest case.

Now consider the second case. Suppose that $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated at AliceChain time $t^* \geq T^{\mathsf{A}}_1$, so pre_a is publicly known after AliceChain time t^*. By assumption, $\mathsf{Normal}^{\mathsf{B}}$ happens, so $P^{\mathsf{B}}_{\mathsf{default}}$ must be activated. In this case, $\mathcal{A}$ has to send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$.

- *Case 1: $\mathcal{A}$ sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before* BobChain *time T^{B}_1.* Since BobChain time T^{B}_1 is earlier than AliceChain time T^{A}_1, pre_s and pre_a are both publicly known at AliceChain time t^*. Thus, during AliceChain time $(t^*, t^* + \tau^{\mathsf{A}}]$, any honest miner will activate $C^{\mathsf{A}}_{\mathsf{burn}}$ if it wins a block. We say $\mathcal{A}$ loses the race if a non-colluding miner mines a new block during AliceChain time $(t^*, t^* + \tau^{\mathsf{A}}]$. Otherwise, we say $\mathcal{A}$ wins the race. If $\mathcal{A}$ loses the race, it gets nothing from $C^{\mathsf{A}}_{\mathsf{refund}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$, and its utility is at most $\$\mathsf{AV}(\ddot{\mathrm{B}}x_b - \ddot{\mathrm{A}}x_a - \ddot{\mathrm{A}}c^{\mathsf{A}}_a)$. Else if $\mathcal{A}$ wins the race, then its utility is at most $\$\mathsf{AV}(\ddot{\mathrm{B}}x_b)$, which can be achieved by activating $P^{\mathsf{A}}_{\mathsf{refund}}$, $C^{\mathsf{A}}_{\mathsf{refund}}$ and $P^{\mathsf{B}}_{\mathsf{default}}$. The probability p that $\mathcal{A}$ wins the race is upper bounded by $p \leq \gamma^{\tau^{\mathsf{A}}}$. Therefore, the expected utility of $\mathcal{A}$ is upper bounded by

$$\$\mathsf{AV}((\ddot{\mathrm{B}}x_b - \ddot{\mathrm{A}}x_a - \ddot{\mathrm{A}}c^{\mathsf{A}}_a) \cdot (1-p) + \ddot{\mathrm{B}}x_b \cdot p). \tag{2}$$

  Since $p \leq \gamma^{\tau^{\mathsf{A}}} \leq \frac{\ddot{\mathrm{A}}c^{\mathsf{A}}_a}{\ddot{\mathrm{A}}c^{\mathsf{A}}_a + \ddot{\mathrm{A}}x_a}$, we have

$$\$\mathsf{AV}((\ddot{\mathrm{B}}x_b - \ddot{\mathrm{A}}x_a - \ddot{\mathrm{A}}c^{\mathsf{A}}_a) \cdot (1-p) + \ddot{\mathrm{B}}x_b \cdot p) < \$\mathsf{AV}(\ddot{\mathrm{B}}x_b - \ddot{\mathrm{A}}x_a).$$

- *Case 2: $\mathcal{A}$ does not send any transaction containing pre_s before* BobChain *time T^{B}_1.* In this case, the honest Bob will send ping to $A^{\mathsf{A}}_{\mathsf{defuse}}$ at BobChain time T^{B}_1. If $A^{\mathsf{A}}_{\mathsf{defuse}}$ has not been activated at AliceChain time $t^* \geq T^{\mathsf{A}}_1$, then, during AliceChain time $(t^*, t^* + \tau^{\mathsf{A}}]$, any honest miner will activate $A^{\mathsf{A}}_{\mathsf{burn}}$ if it wins a block. On the other hand, if $A^{\mathsf{A}}_{\mathsf{defuse}}$ has been activated at AliceChain time $t^* \geq T^{\mathsf{A}}_1$, the honest Bob will send pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$ as soon as $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated. Thus, at AliceChain time $t^* \geq T^{\mathsf{A}}_1$, pre_a and pre_b are both publicly known. Thus, during AliceChain time $(t^*, t^* + \tau^{\mathsf{A}}]$, any honest miner will activate $C^{\mathsf{A}}_{\mathsf{burn}}$ if it wins a block. By the same calculation as the previous case, since $p \leq \gamma^{\tau^{\mathsf{A}}} \leq \frac{\ddot{\mathrm{A}}c^{\mathsf{A}}_a}{\ddot{\mathrm{A}}c^{\mathsf{A}}_a + \ddot{\mathrm{A}}x_a}$, we have $\$\mathsf{AV}((\ddot{\mathrm{A}}x_a - \ddot{\mathrm{A}}c^{\mathsf{A}}_a + \ddot{\mathrm{B}}x_b) \cdot (1-p) + \ddot{\mathrm{B}}x_b \cdot p) < \$\mathsf{AV}(\ddot{\mathrm{B}}x_b - \ddot{\mathrm{A}}x_a)$.

□

Lemma 5.4 (Bob-miner coalition)**.** *Suppose that the hash function $H(\cdot)$ is a one-way function. Let $\mathcal{B}$ be any coalition that consists of Bob and a subset of miners controlling at most γ fraction of mining power. Then, as long as $\gamma^{\tau^{\mathsf{B}}} \leq \frac{\ddot{\mathrm{B}}c_b^{\mathsf{B}}}{\ddot{\mathrm{B}}c_b^{\mathsf{B}}+\ddot{\mathrm{B}}x_b}$, for any PPT coalition strategy $S_{\mathcal{B}}$, except with negligible probability, it must be*

$$\mathsf{util}^{\mathcal{B}}(S_{\mathcal{B}}, HS_{-\mathcal{B}}) \leq \mathsf{util}^{\mathcal{B}}(HS_{\mathcal{B}}, HS_{-\mathcal{B}}),$$

where $HS_{-\mathcal{B}}$ denotes the honest strategy for everyone not in $\mathcal{B}$.

Proof. Recall that the utility of $\mathcal{B}$ is $\$\mathsf{BV}(\ddot{\mathrm{A}}x_a - \ddot{\mathrm{B}}x_b) > 0$ under an honest execution. Now, suppose $\mathcal{B}$ may deviate from the protocol. We may assume that the coalition does not post any new smart contract on the fly and deposit money into it — if it did so, it cannot recover more than its deposit since any player not in $\mathcal{B}$ will not invoke the smart contract. We analyze the two possible cases depending on which phase Alice enters.

Alice enters the abort phase. If $\textsc{Contract}^{\mathrm{A}}$ never becomes active, the net profit of $\mathcal{B}$ from $\mathsf{AliceChain}$ is at most zero. Now, assume $\textsc{Contract}^{\mathrm{A}}$ is active. When Alice enters the abort phase, she never sends any transaction containing pre_s. Ignoring the negligible probability that $\mathcal{B}$ finds pre_s by itself, $P^{\mathsf{A}}_{\mathsf{default}}$ can never be activated, which means $\mathsf{Normal}^{\mathsf{A}}$ never happens. According to Table 2, if $\mathsf{Normal}^{\mathsf{A}}$ does not happens, the net profit of $\mathcal{B}$ from $\mathsf{AliceChain}$ is at most zero. On the other hand, because $\ddot{\mathrm{B}}x_b > \ddot{\mathrm{B}}\epsilon^{\mathsf{B}}$, the net profit of $\mathcal{B}$ from $\mathsf{BobChain}$ is at most zero, no matter $\textsc{Contract}^{\mathrm{B}}$ is active or not.

To sum up, except with negligible probability, the utility of $\mathcal{B}$ is at most zero, which is less than the honest case.

Alice enters the execution phase. By Lemma 5.2, the utility of $\mathcal{B}$ can be more than the honest case only if $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ both happen, so we assume it is the case. Because Alice enters the execution phase, both $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ must be active. Thus, $\mathsf{Refund}^{\mathsf{B}}$ happens only if $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated. When Alice enters the execution phase, she never sends ping to $P^{\mathsf{B}}_{\mathsf{refund}}$, so $P^{\mathsf{B}}_{\mathsf{refund}}$ must be activated by pre_b sent by Bob. Therefore, we may assume that $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated at $\mathsf{BobChain}$ time $t^* \geq T_1^{\mathsf{B}}$, and pre_b is publicly known after $\mathsf{BobChain}$ time t^*. If Alice enters the execution, Bob must have sent pre_c before $\mathsf{BobChain}$ time T_0. Moreover, Alice sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ at $\mathsf{BobChain}$ time T_0 and $T_0 < T_1^{\mathsf{B}}$. Therefore, pre_s, pre_b and pre_c are all publicly known at $\mathsf{BobChain}$ time t^*. Thus, during $\mathsf{BobChain}$ time $(t^*, t^* + \tau^{\mathsf{B}}]$, any honest miner will activate $C^{\mathsf{B}}_{\mathsf{burn}}$ if it wins a block. We say $\mathcal{B}$ loses the race if a non-colluding miner mines a new block during $\mathsf{BobChain}$ time $(t^*, t^* + \tau^{\mathsf{B}}]$. Otherwise, we say $\mathcal{B}$ wins the race. If $\mathcal{B}$ loses the race, it gets nothing from $C^{\mathsf{B}}_{\mathsf{refund}}$ or $C^{\mathsf{B}}_{\mathsf{burn}}$, and its utility is at most $\$\mathsf{BV}(\ddot{\mathrm{A}}x_a - \ddot{\mathrm{B}}x_b - \ddot{\mathrm{B}}c_b^{\mathsf{B}})$ which can be achieved by $P^{\mathsf{A}}_{\mathsf{default}}$. Else if $\mathcal{B}$ wins the race, then its utility is at most $\$\mathsf{BV}(\ddot{\mathrm{A}}x_a)$ which can be achieved by activating $P^{\mathsf{B}}_{\mathsf{refund}}$, $C^{\mathsf{B}}_{\mathsf{refund}}$ and $P^{\mathsf{A}}_{\mathsf{default}}$. Since $p \leq \gamma^{\tau^{\mathsf{B}}} \leq \frac{\ddot{\mathrm{B}}c_b^{\mathsf{B}}}{\ddot{\mathrm{B}}c_b^{\mathsf{B}}+\ddot{\mathrm{B}}x_b}$, we have

$$\$\mathsf{BV}((\ddot{\mathrm{A}}x_a - \ddot{\mathrm{B}}x_b - \ddot{\mathrm{B}}c_b^{\mathsf{B}}) \cdot (1-p) + \ddot{\mathrm{A}}x_a \cdot p) < \$\mathsf{BV}(\ddot{\mathrm{A}}x_a - \ddot{\mathrm{B}}x_b).$$

□

Proof of Theorem 5.1. Now, we are ready to prove Theorem 5.1. In Lemma 5.3 and Lemma 5.4, we show that the atomic swap protocol satisfies γ-CSP-fairness when the coalition consists of Alice or Bob, and possibly with some miners. Because we assume that Alice and Bob are not in the same coalition, it remains to show γ-CSP-fairness when the coalition $\mathcal{C}$ consists only of miners controlling at most γ fraction of the mining power.

Henceforth, we assume Alice and Bob are both honest. It is clear from the protocol that the honest Alice and honest Bob always make the same decision whether to enter the execution phase or abort phase.

Next, when $\mathcal{C}$ follows the protocol, its utility is always zero. Suppose $\mathcal{C}$ may deviate from the protocol. Notice that the utility of $\mathcal{C}$ can be positive only when $A^{\mathsf{A}}_{\mathsf{burn}}$, $B^{\mathsf{B}}_{\mathsf{burn}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$ or $C'^{\mathsf{A}}_{\mathsf{burn}}$ is activated. Because Bob only sends pre_c when $B^{\mathsf{B}}_{\mathsf{defuse}}$ has been activated, ignoring the negligible probability that $\mathcal{C}$ find pre_c by itself, $B^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. In the following, we will show that $A^{\mathsf{A}}_{\mathsf{burn}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$ and $C'^{\mathsf{A}}_{\mathsf{burn}}$ are never activated except with negligible probability. There are two possible cases.

- *Case 1: both Alice and Bob enter the execution phase.* In this case, Alice always sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$, and she never sends any transaction containing pre_a. Ignoring the negligible probability that $\mathcal{C}$ finds pre_a by itself, $C'^{\mathsf{A}}_{\mathsf{burn}}$ can never be activated, and $A^{\mathsf{A}}_{\mathsf{burn}}$ can only be activated by pre_b. Moreover, Alice always sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ at latest at $\mathsf{BobChain}$ time T_0, and thus Bob will not post any transaction containing pre_b. Ignoring the negligible probability that $\mathcal{C}$ finds pre_b by itself, $A^{\mathsf{A}}_{\mathsf{burn}}$ and $C^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. To sum up, except the negligible probability, the utility of $\mathcal{C}$ is at most zero, which is the same as the honest case.
- *Case 2: both Alice and Bob enter the abort phase.* In this case, Bob always sends ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ and Alice always sends ping to $P^{\mathsf{B}}_{\mathsf{refund}}$. Thus, Bob never sends any transaction containing pre_b, and Alice never sends any transaction containing pre_a. Ignoring the negligible probability that $\mathcal{C}$ finds pre_b or pre_a by itself, $A^{\mathsf{A}}_{\mathsf{burn}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$ and $C'^{\mathsf{A}}_{\mathsf{burn}}$ cannot be activated by (pre_a, pre_b). Thus, except with negligible probability, the utility of $\mathcal{C}$ is at most zero, which is the same as the honest case.

5.2 Achieving Bounded Maximin Fairness

Henceforth, let $\alpha \in [0, 1 - 1/\mathsf{poly}(\lambda)]$ denote the maximum fraction of mining power controlled by the set of externally incentivized players, and let $\$E$ be an upper bound on any individual or coalition's valuation of the total possible external incentive.

Proof Roadmap. Conceptually, because Alice and Bob both put the collateral on both $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$, none of them wants to trigger any of the bombs ($B^{\mathsf{B}}_{\mathsf{burn}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $A^{\mathsf{A}}_{\mathsf{burn}}$ and $C'^{\mathsf{A}}_{\mathsf{burn}}$). In Section 5.2.1, we define a set of "bad events" that leads to the activation of the bomb. The bad events are defined such that if any of the bad events is about to happen, it must be that a strategic individual or coalition is about to send a transaction that does not follow from the honest protocol. In Section 5.2.2, we show that whenever the Bob-miner coalition $\mathcal{B}$ is about to send a message that makes a bad event happen, $\mathcal{B}$'s expected utility can be strictly improved if $\mathcal{B}$ simply stops sending any messages (including the message that is about to trigger the bad event) from that moment on (Lemma 5.7). Hence, any strategy that makes the bad event happen is a blatantly irrational strategy for $\mathcal{B}$, so a rational player would never make the bad events happen. Then, we show that as long as none of the bad events happens, the honest Alice's utility is never negative (Lemma 5.6 and Lemma 5.8). A similar argument can be made for the case of an Alice-miner coalition (see Section 5.2.3). Finally, in Section 5.2.4, we combine all the arguments above, and prove that the atomic swap protocol achieves bounded maximin fairness.

5.2.1 Irrational Strategies

In this section, we will define a family of PPT strategies denoted $\overline{\mathcal{R}}$, and we will show given any strategy $S \in \overline{\mathcal{R}}$, we can give a simple modification of S, resulting in a new PPT strategy which

makes the externally incentivized coalition better off — in this sense, the strategy space $\overline{\mathcal{R}}$ is blatantly irrational.

Terminologies. Consider any user-miner coalition, and we define the following terminologies.

- We say an activation point X is *guaranteed to be activated* at some time T, iff either X was already activated before T, or a colluding miner has been chosen as the winning miner at time T, and it activates X in the new block it mines.
- We say a smart contract is *guaranteed to be active* at some time T, iff either the contract was already active before T, or a colluding miner has been chosen as the winning miner at time T, and it includes Alice's and Bob's deposit transactions for the contract in the new block it mines.
- We say Alice is *guaranteed to withdraw her deposit from a contract* at some time T, iff either Alice already withdrew her deposit from the contract before T, or all the following conditions hold.
  - The contract is not active before time T.
  - At time T, a colluding miner is chosen as the winning miner, and Alice's withdrawal transaction is included in the block at time T.

  The case that Bob is guaranteed to withdraw his deposit from the contract is defined similarly.

Irrational strategies. The set $\overline{\mathcal{R}}$ of irrational strategies for the externally incentivized Bob-miner coalition (including Bob alone) $\mathcal{B}$ is the set of strategies such that with non-negligible probability, any of the following happens:

$\mathbf{E}_1$: Before $B^{\mathsf{B}}_{\mathsf{defuse}}$ is guaranteed to be activated and before Bob is guaranteed to withdraw his deposit from $\textsc{Contract}^{\mathrm{B}}$, anyone in $\mathcal{B}$ sends pre_c to ($B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{default}}$ or $C^{\mathsf{B}}_{\mathsf{burn}}$) and the deposit transaction to $\textsc{Contract}^{\mathrm{B}}$.

$\mathbf{E}_2$: $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ are guaranteed to be active before $\mathsf{BobChain}$ time T^{B}. Additionally, anyone in $\mathcal{B}$ sends pre_c to ($B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{default}}$ or $C^{\mathsf{B}}_{\mathsf{burn}}$) before $\mathsf{BobChain}$ time T^{B}_0, and sends pre_b to ($P^{\mathsf{B}}_{\mathsf{refund}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) before ($P^{\mathsf{B}}_{\mathsf{default}}$ or $C^{\mathsf{B}}_{\mathsf{refund}}$) is guaranteed to be activated.

$\mathbf{E}_3$: Before $A^{\mathsf{A}}_{\mathsf{defuse}}$ is guaranteed to be activated and before Bob is guaranteed to withdraw his deposit from $\textsc{Contract}^{\mathrm{A}}$, anyone in $\mathcal{B}$ sends pre_b to ($P^{\mathsf{B}}_{\mathsf{refund}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) and the deposit transaction to $\textsc{Contract}^{\mathrm{A}}$.

$\mathbf{E}_4$: Alice enters the abort phase, and Bob does not send ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ before $\mathsf{AliceChain}$ time T^{A}_1. Additionally, at $\mathsf{AliceChain}$ time T^{A}_1 or later, anyone in $\mathcal{B}$ sends pre_b to ($P^{\mathsf{B}}_{\mathsf{refund}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) before ($P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{refund}}$) is guaranteed to be activated.

The set $\overline{\mathcal{R}}$ of irrational strategies for the externally incentivized Alice-miner coalition (including Alice alone) $\mathcal{A}$ is the set of strategies such that with non-negligible probability, any of the following happens:

$\mathbf{E}_5$: Bob sends pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$ before $\mathsf{BobChain}$ time T^{B}_0, and Alice does not send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ until $\mathsf{BobChain}$ time T^{B}_1. However, at $\mathsf{BobChain}$ time T^{B}_1 or later, anyone in $\mathcal{A}$ sends pre_s to ($P^{\mathsf{B}}_{\mathsf{default}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) before ($P^{\mathsf{B}}_{\mathsf{default}}$ or $C^{\mathsf{B}}_{\mathsf{refund}}$) is guaranteed to be activated.

$\mathbf{E}_6$: Before $A^{\mathsf{A}}_{\mathsf{defuse}}$ is guaranteed to be activated and before Alice is guaranteed to withdraw her deposit from $\textsc{Contract}^{\mathrm{A}}$, anyone in $\mathcal{A}$ sends pre_a to ($A^{\mathsf{A}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{refund}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) and the deposit transaction to $\textsc{Contract}^{\mathrm{A}}$.

$\mathbf{E}_7$: Anyone in $\mathcal{A}$ sends pre_s to ($P^{\mathsf{B}}_{\mathsf{default}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) and pre_a to ($A^{\mathsf{A}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{refund}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) before ($P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{refund}}$) is guaranteed to be activated.

$\mathbf{E}_8$: Any one of the conditions holds.

- Bob enters the execution phase, and Alice does not send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before $\mathsf{BobChain}$ time T^{B}_1. Additionally, at $\mathsf{AliceChain}$ time T^{A}_1 or later, anyone in $\mathcal{A}$ sends pre_a to ($A^{\mathsf{A}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{refund}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) before ($P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{refund}}$) is guaranteed to be activated.
- Bob enters the abort phase, and Alice does not send ping to $P^{\mathsf{B}}_{\mathsf{refund}}$ before $\mathsf{AliceChain}$ time T^{A}_1. Additionally, at $\mathsf{AliceChain}$ time T^{A}_1 or later, anyone in $\mathcal{A}$ sends pre_a to ($A^{\mathsf{A}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{refund}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) before ($P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{refund}}$) is guaranteed to be activated.

The following lemma specifies the upper bounds of the utility of the externally incentivized players.

Lemma 5.5. *Suppose the coalition $\mathcal{B}$ consists of Bob and possibly some miners. Let $\$E$ be an upper bound on $\mathcal{B}$'s valuation of the total possible external incentive. If $\Ddot{\mathbb{B}}c^{\mathsf{B}}_b > \Ddot{\mathbb{B}}\epsilon^{\mathsf{B}}$, $\Ddot{\mathbb{A}}c^{\mathsf{A}}_b > \Ddot{\mathbb{A}}\epsilon^{\mathsf{A}}$, then, the following statements hold.*

- *The utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(\Ddot{\mathbb{A}}x_a) + \$E$.*
- *If $\mathsf{Burn}^{\mathsf{B}}$ is activated by an honest miner $\notin \mathcal{B}$, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(\Ddot{\mathbb{A}}x_a - \Ddot{\mathbb{B}}x_b - \Ddot{\mathbb{B}}c^{\mathsf{B}}_b) + \$E$.*
- *If $\mathsf{Burn}^{\mathsf{A}}$ is activated by an honest miner $\notin \mathcal{B}$, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(-\Ddot{\mathbb{A}}c^{\mathsf{A}}_b) + \$E$.*

Similarly, suppose the coalition $\mathcal{A}$ consists of Alice and possibly some miners. Let $\$E$ be an upper bound on $\mathcal{A}$'s valuation of the total possible external incentive. If $\Ddot{\mathbb{B}}c^{\mathsf{B}}_a > \Ddot{\mathbb{B}}\epsilon^{\mathsf{B}}$, $\Ddot{\mathbb{A}}c^{\mathsf{A}}_a > \Ddot{\mathbb{A}}\epsilon^{\mathsf{A}}$, then, the following statements hold.

- *The utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\Ddot{\mathbb{B}}x_b) + \$E$.*
- *If $\mathsf{Burn}^{\mathsf{B}}$ is activated by an honest miner $\notin \mathcal{A}$, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(-\Ddot{\mathbb{B}}c^{\mathsf{B}}_a) + \$E$.*
- *If $\mathsf{Burn}^{\mathsf{A}}$ is activated by an honest miner $\notin \mathcal{A}$, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\Ddot{\mathbb{B}}x_b - \Ddot{\mathbb{A}}x_a - \Ddot{\mathbb{A}}c^{\mathsf{A}}_a) + \$E$.*

Proof. The maximal possible utilities under different events are summarized in Table 1 and Table 2. The lemma directly follows from the calculation of the utilities in the table. □

5.2.2 Against Externally Incentivized Bob-Miner Coalition

Lemma 5.6. *Let $\mathcal{B}$ be the coalition consisting of Bob and miners controlling no more than α fraction of the mining power where $\alpha \in [0, 1 - 1/\mathsf{poly}(\lambda)]$. Suppose Alice and at least $1 - \alpha$ fraction of mining power are honest. For any PPT strategy by $\mathcal{B}$, except with negligible probability, as long as none of $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3, \mathbf{E}_4$ happens, then, one and only one of the following statement holds.*

1. *$\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ happen in polynomial time.*

2. $\mathsf{Normal}^{\mathsf{B}}$ *and* $\mathsf{Refund}^{\mathsf{A}}$ *happen in polynomial time.*

3. $\mathsf{Refund}^{\mathsf{B}}$ *and* $\mathsf{Refund}^{\mathsf{A}}$ *happen in polynomial time.*

Proof. First, we are going to show that one of $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{B}}$ will happen. There are two cases.

- *Case 1: Alice enters the execution phase.* We will show that $P^{\mathsf{B}}_{\mathsf{default}}$ must be activated in polynomial time except with negligible probability. Once $P^{\mathsf{B}}_{\mathsf{default}}$ is activated, $\mathsf{Normal}^{\mathsf{B}}$ happens. In the execution phase, Alice always sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$. If $B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{refund}}$ and $C^{\mathsf{B}}_{\mathsf{burn}}$ are not activated, the honest miners will include Alice's transaction, pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$, once they mine a block. Thus, it suffices to show that $B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{refund}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$ (all activations points that $P^{\mathsf{B}}_{\mathsf{default}}$ is mutually exclusive with) cannot be activated except with negligible probability. First, because $\mathbf{E}_1$ does not happen, $\mathcal{B}$ never sends pre_c before $B^{\mathsf{B}}_{\mathsf{defuse}}$ is guaranteed to be activated. Thus, $B^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated.

  Next, because Alice enters the execution phase, $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ must be active, and Bob already sent pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$ before $\mathsf{BobChain}$ time T^{B}_0. Because $\mathbf{E}_2$ does not happen, no one in $\mathcal{B}$ sends pre_b to $C^{\mathsf{B}}_{\mathsf{burn}}$ before $P^{\mathsf{B}}_{\mathsf{default}}$ or $C^{\mathsf{B}}_{\mathsf{refund}}$ is guaranteed to be activated. Thus, $C^{\mathsf{B}}_{\mathsf{burn}}$ cannot be activated.

  It remains to show that $P^{\mathsf{B}}_{\mathsf{refund}}$ cannot be activated. In the execution phase, Alice never sends ping to $P^{\mathsf{B}}_{\mathsf{refund}}$, so $P^{\mathsf{B}}_{\mathsf{refund}}$ can be activated only if Bob sends pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$. Because $\mathbf{E}_2$ does not happen, Bob never sends pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$ before $P^{\mathsf{B}}_{\mathsf{default}}$ or $C^{\mathsf{B}}_{\mathsf{refund}}$ is guaranteed to be activated. If $P^{\mathsf{B}}_{\mathsf{default}}$ is guaranteed to be activated, $P^{\mathsf{B}}_{\mathsf{refund}}$ cannot be activated as they are mutually exclusive. On the other hand, if $C^{\mathsf{B}}_{\mathsf{refund}}$ is guaranteed to be activated, $P^{\mathsf{B}}_{\mathsf{refund}}$ must have been activated $\tau^{\mathsf{B}} \geq 1$ $\mathsf{BobChain}$ time before. Thus, by the time $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, $C^{\mathsf{B}}_{\mathsf{refund}}$ has not been guaranteed to be activated. However, and Bob never sends pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$ before $C^{\mathsf{B}}_{\mathsf{refund}}$ is guaranteed to be activated. Therefore, $P^{\mathsf{B}}_{\mathsf{refund}}$ can never be activated.

- *Case 2: Alice enters the abort phase.* In this case, Alice will send the withdrawal transaction to $\textsc{Contract}^{\mathrm{B}}$, and ping to $P^{\mathsf{B}}_{\mathsf{refund}}$. When $\textsc{Contract}^{\mathrm{B}}$ has not been active yet, the honest miner will include Alice's withdrawal transactions once they mine a block. Thus, except with negligible probability, in polynomial time, either $\textsc{Contract}^{\mathrm{B}}$ becomes active, or Alice successfully withdraws her deposit from $\textsc{Contract}^{\mathrm{B}}$. If Alice withdraws her deposit from $\textsc{Contract}^{\mathrm{B}}$, $\mathsf{Refund}^{\mathsf{B}}$ happens.

  Henceforth, we assume $\textsc{Contract}^{\mathrm{B}}$ becomes active. Because $\mathbf{E}_1$ does not happen, $\mathcal{B}$ never sends pre_c before $B^{\mathsf{B}}_{\mathsf{defuse}}$ is guaranteed to be activated. Thus, $B^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. Then, notice that Alice never sends pre_s when she enters the abort phase. Ignoring the negligible probability that $\mathcal{B}$ finds pre_s by itself, $C^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated.

  Thus, since $1-\alpha$ fraction of mining power is honest, either $P^{\mathsf{B}}_{\mathsf{default}}$ or $P^{\mathsf{B}}_{\mathsf{refund}}$ will be activated in polynomial time. If $P^{\mathsf{B}}_{\mathsf{default}}$ is activated, $\mathsf{Normal}^{\mathsf{B}}$ happens. If $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Alice will send ping to $C^{\mathsf{B}}_{\mathsf{refund}}$ when τ^{B} $\mathsf{BobChain}$ time has passed since $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated. Again, the honest miner will include Alice's transaction ping to $C^{\mathsf{B}}_{\mathsf{refund}}$, once they mine a block. Thus, $\mathsf{Refund}^{\mathsf{B}}$ will happen in polynomial time.

Next, we are going to show that one of $\mathsf{Normal}^{\mathsf{A}}$ and $\mathsf{Refund}^{\mathsf{A}}$ will happen. There are two cases.

- *Case 1: Alice enters the execution phase.* As we have shown, when Alice enters the execution phase, $P^{\mathsf{B}}_{\mathsf{default}}$ must be activated in polynomial time. Then, Alice will send ping to $P^{\mathsf{A}}_{\mathsf{default}}$ as soon as $P^{\mathsf{B}}_{\mathsf{default}}$ is activated. Because $\mathbf{E}_3$ does not happen and Alice never sends pre_a when

in the execution phase, $A^{\mathsf{A}}_{\mathsf{burn}}$ cannot be activated. Ignoring the negligible probability that $\mathcal{B}$ finds pre_a by itself, $C^{\mathsf{A}}_{\mathsf{burn}}$ cannot be activated. Because $1-\alpha$ fraction of mining power is honest, either $P^{\mathsf{A}}_{\mathsf{default}}$ or $P^{\mathsf{A}}_{\mathsf{refund}}$ will be activated in polynomial time. If $P^{\mathsf{A}}_{\mathsf{default}}$ is activated, $\mathsf{Normal}^{\mathsf{A}}$ happens, If $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated, Alice will send ping to $C^{\mathsf{A}}_{\mathsf{refund}}$. Again, the honest miner will include a transaction ping to $C^{\mathsf{A}}_{\mathsf{refund}}$, once they mine a block. Thus, $C^{\mathsf{A}}_{\mathsf{refund}}$ will be activated in polynomial time, and $\mathsf{Refund}^{\mathsf{A}}$ happens.

- *Case 2: Alice enters the abort phase.* In this case, Alice will send the withdrawal transaction to $\textsc{Contract}^{\mathrm{A}}$, and ping to $A^{\mathsf{A}}_{\mathsf{defuse}}$. If $\textsc{Contract}^{\mathrm{A}}$ has not been active, the honest miner will include Alice's withdrawal transactions once they mine a block. Thus, in polynomial time, either $\textsc{Contract}^{\mathrm{A}}$ becomes active, or Alice successfully withdraws her deposit from $\textsc{Contract}^{\mathrm{A}}$, so $\mathsf{Refund}^{\mathsf{A}}$ happens.

  Henceforth, we assume $\textsc{Contract}^{\mathrm{A}}$ becomes active in polynomial time. Because $\mathbf{E}_3$ does not happen, $\mathcal{B}$ never sends pre_b before $A^{\mathsf{A}}_{\mathsf{defuse}}$ is guaranteed to be activated. As $A^{\mathsf{A}}_{\mathsf{defuse}}$ and $A^{\mathsf{A}}_{\mathsf{burn}}$ are mutually exclusive, $A^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated via pre_b. Since Alice sends pre_a to $P^{\mathsf{A}}_{\mathsf{refund}}$ only after $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated, 1) $A^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated via pre_a, and 2) $C^{\mathsf{A}}_{\mathsf{burn}}$ can not happen before $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated. If either $P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{refund}}$ are activated, $\mathsf{Normal}^{\mathsf{A}}$ or $\mathsf{Refund}^{\mathsf{A}}$ happens. Otherwise, the honest miner will include the transaction ping to $A^{\mathsf{A}}_{\mathsf{defuse}}$ once they mine a block. Because $1-\alpha$ fraction of mining power is honest, $A^{\mathsf{A}}_{\mathsf{defuse}}$ will be activated in polynomial time.

  When Alice is in abort phase, either Bob sends ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ before $\mathsf{AliceChain}$ time T^{A}_1 or Alice sends pre_a to $P^{\mathsf{A}}_{\mathsf{refund}}$ when $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated. If Bob sends ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ before $\mathsf{AliceChain}$ time T^{A}_1, Alice never sends pre_a, and thus $C^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated. When Alice enters the abort phase, she never sends pre_s, and thus $P^{\mathsf{A}}_{\mathsf{default}}$ can not be activated. As we showed before, $A^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated either, and thus the honest miner will include Bob's transaction, ping to $P^{\mathsf{A}}_{\mathsf{refund}}$, once they mine a block, so $P^{\mathsf{A}}_{\mathsf{refund}}$ will be activated in polynomial time.

  On the other hand, suppose Bob does not send ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ before $\mathsf{AliceChain}$ time T^{A}_1, so Alice sends pre_a to $P^{\mathsf{A}}_{\mathsf{refund}}$. Because $\mathbf{E}_4$ does not happen, $\mathcal{B}$ never sends pre_b before $P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{refund}}$ is guaranteed to be activated. If $P^{\mathsf{A}}_{\mathsf{default}}$ is activated, $\mathsf{Normal}^{\mathsf{A}}$ happens, if $C^{\mathsf{A}}_{\mathsf{refund}}$ is activated, $P^{\mathsf{A}}_{\mathsf{refund}}$ must have been activated previously, and thus $\mathsf{Refund}^{\mathsf{A}}$ happens. Otherwise, recall that Alice never sends pre_s in the abort phase. Without pre_s and pre_b, $P^{\mathsf{A}}_{\mathsf{default}}$ and $C^{\mathsf{A}}_{\mathsf{burn}}$ cannot be activated, and as $A^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated either as shown before, the honest miner will include Alice's transaction, pre_a to $P^{\mathsf{A}}_{\mathsf{refund}}$, once they mine a block. Thus, $P^{\mathsf{A}}_{\mathsf{refund}}$ will again be activated in polynomial time.

  Then, Alice will send ping to $C^{\mathsf{A}}_{\mathsf{refund}}$ when τ^{A} $\mathsf{AliceChain}$ time has passed since $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated. Again, the honest miner will include Alice's transaction, ping to $C^{\mathsf{A}}_{\mathsf{refund}}$, once they mine a block. Thus, $\mathsf{Refund}^{\mathsf{A}}$ will happen in polynomial time.

So far, we have shown one of $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{B}}$ will happen and one of $\mathsf{Normal}^{\mathsf{A}}$ and $\mathsf{Refund}^{\mathsf{A}}$ will happen. To prove the lemma statement, it suffices to show that $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ never happen simultaneously. For the sake of reaching a contradiction, suppose $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ happen. Event $\mathsf{Normal}^{\mathsf{A}}$ happens implies $P^{\mathsf{A}}_{\mathsf{default}}$ is activated, and $P^{\mathsf{A}}_{\mathsf{default}}$ is activated only when Bob sends pre_s or Alice sends ping. The honest Alice only sends ping to $P^{\mathsf{A}}_{\mathsf{default}}$ when $P^{\mathsf{B}}_{\mathsf{default}}$ is activated, which implies $\mathsf{Normal}^{\mathsf{B}}$ happens. Henceforth, we assume $P^{\mathsf{A}}_{\mathsf{default}}$ is activated when by Bob's pre_s. Ignoring the negligible probability that $\mathcal{B}$ can find pre_s by itself, Alice must enter the execution phase and send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ at $\mathsf{BobChain}$ time T^{B}_0. On the other hand, event

$\mathsf{Refund}^{\mathsf{B}}$ happens implies either ($P^{\mathsf{B}}_{\mathsf{refund}} + C^{\mathsf{B}}_{\mathsf{refund}}$) are activated or Bob withdraws his deposit from $\textsc{Contract}^{\mathrm{B}}$. Because Alice enters the execution phase, both $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ must also enter the execution, and thus we exclude that Bob withdraws his deposit from $\textsc{Contract}^{\mathrm{B}}$. Thus, event $\mathsf{Refund}^{\mathsf{B}}$ happens only when ($P^{\mathsf{B}}_{\mathsf{refund}} + C^{\mathsf{B}}_{\mathsf{refund}}$) are activated. Moreover, because Alice enters the execution phase, Bob must have sent pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$. Therefore, to activate $P^{\mathsf{B}}_{\mathsf{refund}}$, Bob has to send pre_b before $C^{\mathsf{B}}_{\mathsf{refund}}$ is guaranteed to be activated, which implies event $\mathbf{E}_2$ happens.

□

Lemma 5.7 (Blatant irrationality of $\overline{\mathcal{R}}$ for Bob-miner coalition)**.** *Suppose that the parameter constraints in Figure 2 hold and that the coalition $\mathcal{B}$ consists of Bob and miners controlling no more than α fraction of the mining power where $\alpha \in [0, 1 - 1/\mathsf{poly}(\lambda)]$. Given any PPT strategy $S_{\mathcal{B}} \in \overline{\mathcal{R}}$ for some (externally incentivized) coalition $\mathcal{B}$, there is a PPT strategy $\widehat{S}_{\mathcal{B}}$ such that*

$$\mathsf{util}^{\mathcal{B}}(\widehat{S}_{\mathcal{B}}, HS_{-\mathcal{B}}) > \mathsf{util}^{\mathcal{B}}(S_{\mathcal{B}}, HS_{-\mathcal{B}}).$$

Proof. Suppose the coalition $\mathcal{B}$ adopts a strategy in which $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$ or $\mathbf{E}_4$ happens with non-negligible probability. We can construct a new strategy for $\mathcal{B}$ with strictly better expected utility. Specifically, consider a modified PPT strategy denoted $\widehat{S}_{\mathcal{B}}$: whenever by the original strategy, the first of $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$ or $\mathbf{E}_4$ is about to happen, $\mathcal{B}$ simply stops sending any messages (including the message that is about to trigger $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$ or $\mathbf{E}_4$) to the contract from that moment on.

By definition, when $\mathbf{E}_1$ or $\mathbf{E}_3$ happens, it must due to a message sent by $\mathcal{B}$. According to the protocol, honest Alice always enters the execution or the abort phase no later than $\mathsf{BobChain}$ time T^{B}_0. Because $\mathsf{AliceChain}$ time T^{A}_1 is later than $\mathsf{BobChain}$ time T^{B}_0 by the choice of the parameters, when $\mathbf{E}_4$ happens, it must also due to a message sent by $\mathcal{B}$. Next, if $\textsc{Contract}^{\mathrm{A}}$ has not been guaranteed to be active, $A^{\mathsf{A}}_{\mathsf{defuse}}$ cannot be guaranteed to be activated. Similarly, if $\textsc{Contract}^{\mathrm{B}}$ has not been guaranteed to be active, $B^{\mathsf{B}}_{\mathsf{defuse}}$ cannot be guaranteed to be activated. Thus, if anyone in $\mathcal{B}$ sends pre_c to ($B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{default}}$ or $C^{\mathsf{B}}_{\mathsf{burn}}$) and sends pre_b to ($P^{\mathsf{B}}_{\mathsf{refund}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$) before $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ both are guaranteed to be active, one of $\mathbf{E}_1$ and $\mathbf{E}_3$ must happen. Consequently, if $\mathbf{E}_1$ and $\mathbf{E}_3$ do not happen while $\mathbf{E}_2$ happens, it must due to a message sent by $\mathcal{B}$. Thus, as long as $\mathcal{B}$ stops sending any messages before the first of $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$ or $\mathbf{E}_4$ is about to happen, none of $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$ and $\mathbf{E}_4$ can happen in the future. By Lemma 5.6, in polynomial time, one of ($\mathsf{Normal}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}}$), ($\mathsf{Normal}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}}$) and ($\mathsf{Refund}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}}$) must happen. By direct calculation, we have the following table. Among ($\mathsf{Normal}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}}$),

	$\mathsf{Normal}^{\mathsf{B}}$	$\mathsf{Refund}^{\mathsf{B}}$	$\mathsf{Burn}^{\mathsf{B}}$
$\mathsf{Normal}^{\mathsf{A}}$	$-\ddot{\mathbb{B}}x_b + \ddot{\mathbb{A}}x_a$	$\ddot{\mathbb{A}}x_a$	$-\ddot{\mathbb{B}}x_b - \ddot{\mathbb{B}}c^{\mathsf{B}}_b + \ddot{\mathbb{A}}x_a$
$\mathsf{Refund}^{\mathsf{A}}$	$-\ddot{\mathbb{B}}x_b$	0	$-\ddot{\mathbb{B}}x_b - \ddot{\mathbb{B}}c^{\mathsf{B}}_b$
$\mathsf{Burn}^{\mathsf{A}}$	$-\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}c^{\mathsf{A}}_b$	$-\ddot{\mathbb{A}}c^{\mathsf{A}}_b$	$-\ddot{\mathbb{B}}x_b - \ddot{\mathbb{B}}c^{\mathsf{B}}_b - \ddot{\mathbb{A}}c^{\mathsf{A}}_b$

Table 3: Bob's net profit under all possible events.

($\mathsf{Normal}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}}$) and ($\mathsf{Refund}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}}$), $\mathcal{B}$'s utility is at least $\$\mathsf{BV}(-\ddot{\mathbb{B}}x_b)$. In other words, we have $\mathsf{util}^{\mathcal{B}}(\widehat{S}_{\mathcal{B}}, HS_{-\mathcal{B}}) \geq \$\mathsf{BV}(-\ddot{\mathbb{B}}x_b)$. Thus, to show $\mathsf{util}^{\mathcal{B}}(\widehat{S}_{\mathcal{B}}, HS_{-\mathcal{B}}) \geq \mathsf{util}^{\mathcal{B}}(S_{\mathcal{B}}, HS_{-\mathcal{B}})$, we only need to show $\mathsf{util}^{\mathcal{B}}(S_{\mathcal{B}}, HS_{-\mathcal{B}}) < \$\mathsf{BV}(-\ddot{\mathbb{B}}x_b)$.

We consider four cases, depending on whether $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$ or $\mathbf{E}_4$ happens first in the original strategy $S_{\mathcal{B}}$.

Event $\mathbf{E}_1$ happens first. Consider some strategy $S_1 \in \overline{\mathcal{R}}$ with non-negligible probability that $\mathbf{E}_1$ happens. When $\mathbf{E}_1$ happens, because Bob is not guaranteed to withdraw his deposit from

$\textsc{Contract}^{\mathsf{B}}$, the $1-\alpha$ fraction of honest miners would send the deposit to $\textsc{Contract}^{\mathsf{B}}$ and pre_c to $B^{\mathsf{B}}_{\mathsf{burn}}$ (and potentially $C^{\mathsf{B}}_{\mathsf{burn}}$ as well if pre_s and pre_b are available), if they are chosen to mine a block. By Lemma 5.5, if $B^{\mathsf{B}}_{\mathsf{burn}}$ or $C^{\mathsf{B}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(\ddot{\mathbb{A}}x_a - \ddot{\mathbb{B}}x_b - \ddot{\mathbb{B}}c_b^{\mathsf{B}}) + \$E$. On the other hand, if neither $B^{\mathsf{B}}_{\mathsf{burn}}$ nor $C^{\mathsf{B}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(\ddot{\mathbb{A}}x_a) + \$E$. When $\mathbf{E}_1$ happens, the probability that $B^{\mathsf{B}}_{\mathsf{burn}}$ or $C^{\mathsf{B}}_{\mathsf{burn}}$ is activated by an honest miner is at least $1-\alpha$. Thus, the utility of $\mathcal{B}$ is at most

$$(1-\alpha)(\$\mathsf{BV}(\ddot{\mathbb{A}}x_a - \ddot{\mathbb{B}}x_b - \ddot{\mathbb{B}}c_b^{\mathsf{B}}) + \$E) + \alpha(\$\mathsf{BV}(\ddot{\mathbb{A}}x_a) + \$E) < \$\mathsf{BV}(-\ddot{\mathbb{B}}x_b),$$

where the inequality arises from the fact that $\$\mathsf{BV}(\ddot{\mathbb{B}}c_b^{\mathsf{B}}) > \frac{\$\mathsf{BV}(\ddot{\mathbb{A}}x_a + \alpha\ddot{\mathbb{B}}x_b) + \$E}{1-\alpha}$.

Event $\mathbf{E}_2$ happens first. Consider some strategy $S_2 \in \overline{\mathcal{R}}$ with non-negligible probability that $\mathbf{E}_2$ happens. When $\mathbf{E}_2$ happens, because both $\textsc{Contract}^{\mathsf{B}}$ and $\textsc{Contract}^{\mathsf{A}}$ become active before $\mathsf{BobChain}$ time T^{B} and Bob sends pre_c before $\mathsf{BobChain}$ time T_0^{B}, Alice enters the execution phase. In this case, Alice always sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ at $\mathsf{BobChain}$ time T_0^{B}. However, Bob also sends pre_b before $C^{\mathsf{B}}_{\mathsf{refund}}$ is guaranteed to be activated. Thus, the $1-\alpha$ fraction of honest miners would send (pre_s, pre_b, pre_c) to $C^{\mathsf{B}}_{\mathsf{burn}}$ (and potentially send pre_c to $B^{\mathsf{B}}_{\mathsf{burn}}$), if they are chosen to mine a block. By Lemma 5.5, if $C^{\mathsf{B}}_{\mathsf{burn}}$ or $B^{\mathsf{B}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(\ddot{\mathbb{A}}x_a - \ddot{\mathbb{B}}x_b - \ddot{\mathbb{B}}c_b^{\mathsf{B}}) + \$E$. On the other hand, if neither $C^{\mathsf{B}}_{\mathsf{burn}}$ nor $B^{\mathsf{B}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(\ddot{\mathbb{A}}x_a) + \$E$. When $\mathbf{E}_2$ happens, the probability that $C^{\mathsf{B}}_{\mathsf{burn}}$ or $B^{\mathsf{B}}_{\mathsf{burn}}$ is activated by an honest miner is at least $1-\alpha$. Thus, by the same calculation as the previous case, the utility of $\mathcal{B}$ is strictly less than $\$\mathsf{BV}(-\ddot{\mathbb{B}}x_b)$.

Event $\mathbf{E}_3$ happens first. Consider some strategy $S_3 \in \overline{\mathcal{R}}$ with non-negligible probability that $\mathbf{E}_3$ happens. When $\mathbf{E}_3$ happens, because Bob is not guaranteed to withdraw his deposit from $\textsc{Contract}^{\mathsf{A}}$, the $1-\alpha$ fraction of honest miners would send the deposit to $\textsc{Contract}^{\mathsf{A}}$ and pre_b to $A^{\mathsf{A}}_{\mathsf{burn}}$ (and to $C^{\mathsf{A}}_{\mathsf{burn}}$ if pre_a is available as well), if they are chosen to mine a block. By Lemma 5.5, if $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(-\ddot{\mathbb{A}}c_b^{\mathsf{A}}) + \$E$. On the other hand, if neither $A^{\mathsf{A}}_{\mathsf{burn}}$ nor $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(\ddot{\mathbb{A}}x_a) + \$E$. When $\mathbf{E}_3$ happens, the probability that $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner is at least $1-\alpha$. Thus, the utility of $\mathcal{B}$ is at most

$$(1-\alpha)(\$\mathsf{BV}(-\ddot{\mathbb{A}}c_b^{\mathsf{A}}) + \$E) + \alpha(\$\mathsf{BV}(\ddot{\mathbb{A}}x_a) + \$E) < \$\mathsf{BV}(-\ddot{\mathbb{B}}x_b),$$

where the inequality arises from the fact that $\$\mathsf{BV}(\ddot{\mathbb{A}}c_b^{\mathsf{A}}) > \frac{\$\mathsf{BV}(\ddot{\mathbb{B}}x_b + \alpha\ddot{\mathbb{A}}x_a) + \$E}{1-\alpha}$.

Event $\mathbf{E}_4$ happens first. Consider some strategy $S_4 \in \overline{\mathcal{R}}$ with non-negligible probability that $\mathbf{E}_4$ happens. When $\mathbf{E}_4$ happens, because Alice enters the abort phase and Bob does not send ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ before $\mathsf{AliceChain}$ time T_1^{A}, Alice will send pre_a as soon as $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated. However, Bob also sends pre_b before $P^{\mathsf{A}}_{\mathsf{default}}$ or $C^{\mathsf{A}}_{\mathsf{refund}}$ is guaranteed to be activated. If Bob sends pre_b before $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated, the $1-\alpha$ fraction of honest miners would send pre_b to $A^{\mathsf{A}}_{\mathsf{burn}}$, if they are chosen to mine a block. If Bob sends pre_b after $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated, the $1-\alpha$ fraction of honest miners would send (pre_a, pre_b) to $C^{\mathsf{A}}_{\mathsf{burn}}$, if they are chosen to mine a block. By Lemma 5.5, if $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(-\ddot{\mathbb{A}}c_b^{\mathsf{A}}) + \$E$. On the other hand, if neither $A^{\mathsf{A}}_{\mathsf{burn}}$ nor $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{B}$ is at most $\$\mathsf{BV}(\ddot{\mathbb{A}}x_a) + \$E$. When $\mathbf{E}_4$ happens, the probability that $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner is at least $1-\alpha$. Thus, by the same calculation as the previous case, the utility of $\mathcal{B}$ is strictly less than $\$\mathsf{BV}(-\ddot{\mathbb{B}}x_b)$.

Finally, notice that the above analysis holds even if $\mathcal{B}$ may post a new contract on the fly during the protocol execution, since all other players are honest and will not deposit money into the new contract. □

Lemma 5.8 (Against Externally Incentivized Bob-Miner Coalition)**.** *Suppose that the hash function $H(\cdot)$ is a one-way function. Let $\mathcal{C}$ be a group consisting of Alice and possibly any subset of the miners, and let $\mathcal{B}$ be a disjoint coalition consisting of Bob and at most α fraction of the mining power where $\alpha \in [0, 1 - 1/\mathsf{poly}(\lambda)]$. Suppose that $\mathcal{C}$ does not have external incentives but $\mathcal{B}$ may have up to $\$E$ amount of external incentives. Let $S_{\mathcal{B}}$ be an arbitrary PPT strategy of $\mathcal{B}$ that is not in $\overline{\mathcal{R}}$. Then, there exists a negligible function $\mathsf{negl}(\cdot)$ such that except with negligible probability $\mathsf{negl}(\lambda)$, it holds that*

$$\mathsf{util}^{\mathcal{C}}(HS_{\mathcal{C}}, S_{\mathcal{B}}, HS_{\mathcal{D}}) \geq 0,$$

where $\mathcal{D}$ denotes everyone else not in $\mathcal{C} \cup \mathcal{B}$.

Proof. By Lemma 5.7, any strategy that makes one of $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3, \mathbf{E}_4$ happen is blatantly irrational. By direct calculation, we have the following table.

	$\mathsf{Normal}^{\mathsf{B}}$	$\mathsf{Refund}^{\mathsf{B}}$	$\mathsf{Burn}^{\mathsf{B}}$
$\mathsf{Normal}^{\mathsf{A}}$	$\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a$	$-\ddot{\mathbb{A}}x_a$	$-\ddot{\mathbb{B}}c_a^{\mathsf{B}} - \ddot{\mathbb{A}}x_a$
$\mathsf{Refund}^{\mathsf{A}}$	$\ddot{\mathbb{B}}x_b$	0	$-\ddot{\mathbb{B}}c_a^{\mathsf{B}}$
$\mathsf{Burn}^{\mathsf{A}}$	$\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a - \ddot{\mathbb{A}}c_a^{\mathsf{A}}$	$-\ddot{\mathbb{A}}x_a - \ddot{\mathbb{A}}c_a^{\mathsf{A}}$	$-\ddot{\mathbb{B}}c_a^{\mathsf{B}} - \ddot{\mathbb{A}}x_a - \ddot{\mathbb{A}}c_a^{\mathsf{A}}$

Table 4: Alice's net profit under all possible events.

By Lemma 5.6, if none of $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3, \mathbf{E}_4$ happens, then one of ($\mathsf{Normal}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}}$), ($\mathsf{Normal}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}}$) and ($\mathsf{Refund}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}}$) must happen. Because $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a) > 0$, except with some negligible probability, for all three possible cases Alice's utility is non-negative.

□

5.2.3 Against Externally Incentivized Alice-Miner Coalition

Lemma 5.9. *Let $\mathcal{A}$ be the coalition consisting of Alice and miners controlling no more than α fraction of the mining power where $\alpha \in [0, 1 - 1/\mathsf{poly}(\lambda)]$. Suppose Bob and at least $1 - \alpha$ fraction of mining power are honest. For any PPT strategy by $\mathcal{A}$, except with negligible probability, as long as none of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7, \mathbf{E}_8$ happens, then, one and only one of the following statement holds.*

1. $\mathsf{Normal}^{\mathsf{B}}$ *and* $\mathsf{Normal}^{\mathsf{A}}$ *happen in polynomial time.*
2. $\mathsf{Refund}^{\mathsf{B}}$ *and* $\mathsf{Normal}^{\mathsf{A}}$ *happen in polynomial time.*
3. $\mathsf{Refund}^{\mathsf{B}}$ *and* $\mathsf{Refund}^{\mathsf{A}}$ *happen in polynomial time.*

Proof. First, we are going to show that one of $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{B}}$ will happen. There are two cases.

- *Case 1: Bob enters the execution phase.* In this case, $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ must be active, and Bob already sent pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$ before $\mathsf{BobChain}$ time T_0^{B}. There are two subcases.

– *Subcase 1: Alice sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before* BobChain *time T^{B}_1.* In this case, Bob never sends pre_b. Without pre_b, $C^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. As Bob is in the execution phase, $B^{\mathsf{B}}_{\mathsf{defuse}}$ was activated and thus $B^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. As Alice sent pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ and Bob sent pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$, either $P^{\mathsf{B}}_{\mathsf{default}}$ is activated in polynomial time, or $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated instead. If $P^{\mathsf{B}}_{\mathsf{default}}$ is activated, $\mathsf{Normal}^{\mathsf{B}}$ happens. If $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Bob sends an empty message to $C^{\mathsf{B}}_{\mathsf{refund}}$, and thus $C^{\mathsf{B}}_{\mathsf{refund}}$ is activated in polynomial time and $\mathsf{Refund}^{\mathsf{B}}$ happens.

– *Subcase 2: Alice does not send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before* BobChain *time T^{B}_1.* In this case, Bob will send ping to $A^{\mathsf{A}}_{\mathsf{defuse}}$. As $\mathbf{E}_6$ does not happen, Alice never sends pre_a before $A^{\mathsf{A}}_{\mathsf{defuse}}$ is guaranteed to be activated, and thus $A^{\mathsf{A}}_{\mathsf{burn}}$ is never activated. Thus, $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated in polynomial time and Bob sends pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$ as soon as $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated. Because $\mathbf{E}_5$ does not happen, Alice never sends pre_s before $P^{\mathsf{B}}_{\mathsf{default}}$ or $C^{\mathsf{B}}_{\mathsf{refund}}$ is activated. Thus, $C^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. As Bob is in the execution phase, $B^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. Thus, either $P^{\mathsf{B}}_{\mathsf{default}}$ is activated ($\mathsf{Normal}^{\mathsf{B}}$ happens), or $C^{\mathsf{B}}_{\mathsf{refund}}$ is activated (thus $P^{\mathsf{B}}_{\mathsf{refund}}$ has been activated before and $\mathsf{Refund}^{\mathsf{B}}$ happens), or the honest miner will include Bob's transaction, pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$, once they mine a block. Then, when τ^{B} BobChain time has passed since $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Bob will send ping to $C^{\mathsf{B}}_{\mathsf{refund}}$ and thus $\mathsf{Refund}^{\mathsf{B}}$ happens.

- *Case 2: Bob enters the abort phase.* In this case, Bob will send the withdrawal transaction to $\textsc{Contract}^{\mathrm{B}}$, and ping to $A^{\mathsf{A}}_{\mathsf{defuse}}$. If $\textsc{Contract}^{\mathrm{B}}$ has not been active, the honest miner will include Bob's withdrawal transactions once they mine a block. Thus, in polynomial time, either $\textsc{Contract}^{\mathrm{B}}$ becomes active, or Bob successfully withdraws his deposit from $\textsc{Contract}^{\mathrm{B}}$, which implies $\mathsf{Refund}^{\mathsf{B}}$ happens.

  Henceforth, we assume $\textsc{Contract}^{\mathrm{B}}$ becomes active in polynomial time. Because Bob is in the abort phase, Bob never sent pre_c and thus, up to negligible probability, $B^{\mathsf{B}}_{\mathsf{burn}}$ and $C^{\mathsf{B}}_{\mathsf{burn}}$ can not be activated. Because $\mathbf{E}_6$ does not happen, $\mathcal{A}$ never sends pre_a before $A^{\mathsf{A}}_{\mathsf{defuse}}$ is guaranteed to be activated. Thus, $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated in polynomial time. Then, either Alice sends ping to $P^{\mathsf{B}}_{\mathsf{refund}}$ before AliceChain time T^{A}_1, or Bob sends pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$. Either way, either $P^{\mathsf{B}}_{\mathsf{default}}$ is activated (thus $\mathsf{Normal}^{\mathsf{B}}$ happens), or $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated in polynomial time. When τ^{B} BobChain time has passed since $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Bob will send ping to $C^{\mathsf{B}}_{\mathsf{refund}}$. Thus, $C^{\mathsf{B}}_{\mathsf{refund}}$ will be activated, and so $\mathsf{Refund}^{\mathsf{B}}$ happens.

Next, we are going to show that one of $\mathsf{Normal}^{\mathsf{A}}$ and $\mathsf{Refund}^{\mathsf{A}}$ will happen. There are two cases.

- *Case 1: Bob enters the execution phase.* First, because $\mathbf{E}_6$ does not happen, Alice never sends pre_a before $A^{\mathsf{A}}_{\mathsf{defuse}}$ is guaranteed to be activated, and as Bob sends pre_b only after $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated, it follows that $A^{\mathsf{A}}_{\mathsf{burn}}$ is never activated.

  Suppose Alice sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before BobChain time T^{B}_1. Then, Bob never sends pre_b, and so $C^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated via (pre_a, pre_b). Because $\mathbf{E}_7$ does not happen, $C^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated via (pre_s, pre_a). Thus, $C^{\mathsf{A}}_{\mathsf{burn}}$ is never activated. As Bob sends pre_s to $P^{\mathsf{A}}_{\mathsf{default}}$, either $\mathsf{Normal}^{\mathsf{A}}$ happens, or $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated. In the latter case, Bob sends ping to $C^{\mathsf{A}}_{\mathsf{refund}}$, and so $\mathsf{Refund}^{\mathsf{A}}$ happens.

  Henceforth, we assume Alice does not send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before BobChain time T^{B}_1. As we have shown before, either $\mathsf{Normal}^{\mathsf{B}}$ happens, or $\mathsf{Refund}^{\mathsf{B}}$ happens. If $\mathsf{Normal}^{\mathsf{B}}$ happens, Bob sends pre_s to $P^{\mathsf{A}}_{\mathsf{default}}$. If $\mathsf{Refund}^{\mathsf{B}}$ happens, Bob will send ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ as soon as $C^{\mathsf{B}}_{\mathsf{refund}}$ is activated. As $\mathbf{E}_8$ does not happen, $C^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated using pre_a after AliceChain time T^{A}_1. As $\mathbf{E}_6$ does not happen, $C^{\mathsf{A}}_{\mathsf{burn}}$ can not be activated using pre_a before AliceChain

time T_1^{A} either. Thus, $C_{\mathsf{burn}}^{\mathsf{A}}$ is never activated. Thus, in both cases ($\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{B}}$), either $P_{\mathsf{default}}^{\mathsf{A}}$ or $P_{\mathsf{refund}}^{\mathsf{A}}$ will be activated in polynomial time. If $P_{\mathsf{default}}^{\mathsf{A}}$ is activated, $\mathsf{Normal}^{\mathsf{A}}$ happens. If $P_{\mathsf{refund}}^{\mathsf{A}}$ is activated, Bob will send ping to $C_{\mathsf{refund}}^{\mathsf{A}}$. The honest miners will include Bob's transaction, ping to $C_{\mathsf{refund}}^{\mathsf{A}}$, once they mine a block. Thus, $C_{\mathsf{refund}}^{\mathsf{A}}$ will be activated in polynomial time, so $\mathsf{Refund}^{\mathsf{A}}$ happens.

- *Case 2: Bob enters the abort phase.* In this case, Bob will send the withdrawal transaction to $\textsc{Contract}^{\mathrm{A}}$, ping to $P_{\mathsf{refund}}^{\mathsf{A}}$ and ping to $A_{\mathsf{defuse}}^{\mathsf{A}}$. If $\textsc{Contract}^{\mathrm{A}}$ has not been active, the honest miner will include Bob's withdrawal transactions once they mine a block. Thus, in polynomial time, either $\textsc{Contract}^{\mathrm{A}}$ becomes active, or Bob successfully withdraws his deposit from $\textsc{Contract}^{\mathrm{A}}$, which implies $\mathsf{Refund}^{\mathsf{A}}$ happens.

  Henceforth, we assume $\textsc{Contract}^{\mathrm{A}}$ becomes active in polynomial time. Because $\mathbf{E}_6$ does not happen, $\mathcal{A}$ never sends pre_a before $A_{\mathsf{defuse}}^{\mathsf{A}}$ is guaranteed to be activated. As Bob's sends ping to $A_{\mathsf{defuse}}^{\mathsf{A}}$, $A_{\mathsf{defuse}}^{\mathsf{A}}$ will be activated in polynomial time. If Alice sends ping to $P_{\mathsf{refund}}^{\mathsf{B}}$ by $\mathsf{AliceChain}$ time T_1^{A}, Bob never sends pre_b, and thus $C_{\mathsf{burn}}^{\mathsf{A}}$ can be activated only via (pre_s, pre_a) (excluded by $\mathbf{E}_7$). If Alice does not send ping to $P_{\mathsf{refund}}^{\mathsf{B}}$ by $\mathsf{AliceChain}$ time T_1^{A}, because $\mathbf{E}_8$ does not happen, Alice does not send pre_a before $P_{\mathsf{default}}^{\mathsf{A}}$ or $C_{\mathsf{refund}}^{\mathsf{A}}$ is guaranteed to be activated. Thus, $C_{\mathsf{burn}}^{\mathsf{A}}$ can not be activated via pre_a. Thus, $C_{\mathsf{burn}}^{\mathsf{A}}$ is never activated. As Bob sends ping to $P_{\mathsf{refund}}^{\mathsf{A}}$, or Alice's transaction, either $P_{\mathsf{default}}^{\mathsf{A}}$ or $P_{\mathsf{refund}}^{\mathsf{A}}$ will be activated in polynomial time. If $P_{\mathsf{default}}^{\mathsf{A}}$ is activated, $\mathsf{Normal}^{\mathsf{A}}$ happens. If $P_{\mathsf{refund}}^{\mathsf{A}}$ is activated, Bob will send ping to $C_{\mathsf{refund}}^{\mathsf{A}}$. Thus, $C_{\mathsf{refund}}^{\mathsf{A}}$ will be activated in polynomial time, so $\mathsf{Refund}^{\mathsf{A}}$ happens.

So far, we have shown one of $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{B}}$ will happen and one of $\mathsf{Normal}^{\mathsf{A}}$ and $\mathsf{Refund}^{\mathsf{A}}$ will happen. To prove the lemma statement, it suffices to show that $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{A}}$ never happen simultaneously. For the sake of reaching a contradiction, suppose $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{A}}$ happen. Event $\mathsf{Normal}^{\mathsf{B}}$ happens implies $P_{\mathsf{default}}^{\mathsf{B}}$ is activated. If Bob enters the abort phase, Bob never sends pre_c to $P_{\mathsf{default}}^{\mathsf{B}}$. Ignoring the negligible probability that $\mathcal{A}$ finds pre_c by itself and forges Bob's signature, Bob must enter the execution, which implies both $\textsc{Contract}^{\mathrm{B}}$ and $\textsc{Contract}^{\mathrm{A}}$ are active. On the other hand, event $\mathsf{Refund}^{\mathsf{A}}$ happens implies either $(P_{\mathsf{refund}}^{\mathsf{A}} + C_{\mathsf{refund}}^{\mathsf{A}})$ are activated or Alice withdraws her deposit from $\textsc{Contract}^{\mathrm{A}}$. As we have shown, $\textsc{Contract}^{\mathrm{A}}$ must be active, so $\mathsf{Refund}^{\mathsf{A}}$ happens implies $(P_{\mathsf{refund}}^{\mathsf{A}} + C_{\mathsf{refund}}^{\mathsf{A}})$ are activated. In the execution phase, Bob will send ping to $P_{\mathsf{refund}}^{\mathsf{A}}$ only if $C_{\mathsf{refund}}^{\mathsf{B}}$ is activated, which is impossible given that $P_{\mathsf{default}}^{\mathsf{B}}$ is activated. Thus, $P_{\mathsf{refund}}^{\mathsf{A}}$ can be activated only if Alice sends pre_a to $P_{\mathsf{refund}}^{\mathsf{A}}$. Because $C_{\mathsf{refund}}^{\mathsf{A}}$ can be activated when τ^{A} $\mathsf{AliceChain}$ time has passed since $P_{\mathsf{refund}}^{\mathsf{A}}$ is activated, Alice must send pre_a to $P_{\mathsf{refund}}^{\mathsf{A}}$ before $C_{\mathsf{refund}}^{\mathsf{A}}$ is guaranteed to be activated. There are two subcases.

- *Subcase 1: pre_s is sent to $P_{\mathsf{default}}^{\mathsf{B}}$ before $\mathsf{BobChain}$ time T_1^{B}.* Because $P_{\mathsf{refund}}^{\mathsf{A}}$ must be activated after $\mathsf{AliceChain}$ time T_1^{A} which is later than $\mathsf{BobChain}$ time T_1^{B}, $\mathcal{A}$ must send pre_s to $P_{\mathsf{default}}^{\mathsf{B}}$ and pre_a to $P_{\mathsf{refund}}^{\mathsf{A}}$ before $C_{\mathsf{refund}}^{\mathsf{A}}$ is guaranteed to be activated. Thus, depends on whether $A_{\mathsf{defuse}}^{\mathsf{A}}$ is guaranteed to be activated, either $\mathbf{E}_6$ or $\mathbf{E}_7$ happens.

- *Subcase 2: pre_s has not been sent to $P_{\mathsf{default}}^{\mathsf{B}}$ before $\mathsf{BobChain}$ time T_1^{B}.* Because Bob enters the execution phase, he will send pre_b to $P_{\mathsf{refund}}^{\mathsf{B}}$ as soon as $A_{\mathsf{defuse}}^{\mathsf{A}}$ is activated. Thus, depending on whether $A_{\mathsf{defuse}}^{\mathsf{A}}$ is guaranteed to be activated, either $\mathbf{E}_6$ or $\mathbf{E}_8$ happens.

Because we assume none of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7, \mathbf{E}_8$ happens, either subcase leads to a contradiction.

□

Lemma 5.10 (Blatant irrationality of $\overline{\mathcal{R}}$ for Alice-miner coalition)**.** *Suppose that the parameter constraints in Figure 2 hold and that the coalition $\mathcal{A}$ consists of Alice and miners controlling no*

more than α fraction of the mining power where $\alpha \in [0, 1 - 1/\mathsf{poly}(\lambda)]$. Given any PPT strategy $S_{\mathcal{A}} \in \overline{\mathcal{R}}$ for some (externally incentivized) coalition $\mathcal{A}$, there is a PPT strategy $\widehat{S}_{\mathcal{A}}$ such that

$$\mathsf{util}^{\mathcal{A}}(\widehat{S}_{\mathcal{A}}, HS_{-\mathcal{A}}) > \mathsf{util}^{\mathcal{A}}(S_{\mathcal{A}}, HS_{-\mathcal{A}}).$$

Proof. Suppose the coalition $\mathcal{A}$ adopts a strategy in which $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7$ or $\mathbf{E}_8$ happens with non-negligible probability. We can construct a new strategy for $\mathcal{A}$ with strictly better expected utility. Specifically, consider a modified PPT strategy denoted $\widehat{S}_{\mathcal{A}}$: whenever by the original strategy, the first of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7$ or $\mathbf{E}_8$ is about to happen, $\mathcal{A}$ simply stops sending any messages (including the message that is about to trigger $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7$ or $\mathbf{E}_8$) to the contract from that moment on.

By definition, when $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7$ or $\mathbf{E}_8$ happens, it must be caused by a message sent by $\mathcal{A}$. Thus, as long as $\mathcal{A}$ stops sending any messages before the first of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7$ or $\mathbf{E}_8$ is about to happen, none of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7$ and $\mathbf{E}_8$ can happen in the future. By Lemma 5.9, in polynomial time, one of $(\mathsf{Normal}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}})$, $(\mathsf{Refund}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}})$ and $(\mathsf{Refund}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}})$ must happen. According to Table 4, among $(\mathsf{Normal}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}})$, $(\mathsf{Refund}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}})$ and $(\mathsf{Refund}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}})$, $\mathcal{A}$'s utility is at least $\$\mathsf{AV}(-\text{Å}x_a)$. In other words, we have $\mathsf{util}^{\mathcal{A}}(\widehat{S}_{\mathcal{A}}, HS_{-\mathcal{A}}) \geq \$\mathsf{AV}(-\text{Å}x_a)$. Thus, to show $\mathsf{util}^{\mathcal{A}}(\widehat{S}_{\mathcal{A}}, HS_{-\mathcal{A}}) \geq \mathsf{util}^{\mathcal{A}}(S_{\mathcal{A}}, HS_{-\mathcal{A}})$, we only need to show $\mathsf{util}^{\mathcal{A}}(S_{\mathcal{A}}, HS_{-\mathcal{A}}) < \$\mathsf{AV}(-\text{Å}x_a)$.

We consider four cases, depending on whether $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7$ or $\mathbf{E}_8$ happens first in the original strategy $S_{\mathcal{A}}$.

Event $\mathbf{E}_5$ happens first. Consider some strategy $S_5 \in \overline{\mathcal{R}}$ with non-negligible probability that $\mathbf{E}_5$ happens. When $\mathbf{E}_5$ happens, the honest Bob will send pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$ as soon as $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated. Thus, when $\mathcal{A}$ sends any transaction containing pre_s, the $1-\alpha$ fraction of honest miners would send (pre_s, pre_b, pre_c) to $C^{\mathsf{B}}_{\mathsf{burn}}$, if they are chosen to mine a block. By Lemma 5.5, if $C^{\mathsf{B}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(-\text{Ƀ}c^{\mathsf{B}}_a) + \$E$. On the other hand, if $C^{\mathsf{B}}_{\mathsf{burn}}$ is not activated by an honest miner, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\text{Ƀ}x_b) + \$E$. When $\mathbf{E}_5$ happens, the probability that $C^{\mathsf{B}}_{\mathsf{burn}}$ is activated by an honest miner is at least $1 - \alpha$. Thus, the utility of $\mathcal{A}$ is at most

$$(1-\alpha)(\$\mathsf{AV}(-\text{Ƀ}c^{\mathsf{B}}_a) + \$E) + \alpha(\$\mathsf{AV}(\text{Ƀ}x_b) + \$E) < \$\mathsf{AV}(-\text{Å}x_a),$$

where the inequality arises from the fact that $\$\mathsf{AV}(\text{Ƀ}c^{\mathsf{B}}_a) > \frac{\$\mathsf{AV}(\text{Å}x_a + \alpha\text{Ƀ}x_b) + \$E}{1-\alpha}$.

Event $\mathbf{E}_6$ happens first. Consider some strategy $S_6 \in \overline{\mathcal{R}}$ with non-negligible probability that $\mathbf{E}_6$ happens. When $\mathbf{E}_6$ happens, the $1 - \alpha$ fraction of honest miners would send pre_a to $A^{\mathsf{A}}_{\mathsf{burn}}$ (or to $C^{\mathsf{A}}_{\mathsf{burn}}$ if additionally either pre_s or pre_b is known), if they are chosen to mine a block. By Lemma 5.5, if either $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\text{Ƀ}x_b - \text{Å}x_a - \text{Å}c^{\mathsf{A}}_a) + \$E$. On the other hand, if neither $A^{\mathsf{A}}_{\mathsf{burn}}$ nor $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\text{Ƀ}x_b) + \$E$. When $\mathbf{E}_6$ happens, the probability that $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner is at least $1 - \alpha$. Thus, the utility of $\mathcal{A}$ is at most

$$(1-\alpha)(\$\mathsf{AV}(\text{Ƀ}x_b - \text{Å}x_a - \text{Å}c^{\mathsf{A}}_a) + \$E) + \alpha(\$\mathsf{AV}(\text{Ƀ}x_b) + \$E) < \$\mathsf{AV}(-\text{Å}x_a),$$

where the inequality arises from the fact that $\$\mathsf{AV}(\text{Å}c^{\mathsf{A}}_a) > \frac{\$\mathsf{AV}(\text{Ƀ}x_b + \alpha\text{Å}x_a) + \$E}{1-\alpha}$.

Event $\mathbf{E}_7$ happens first. Consider some strategy $S_7 \in \overline{\mathcal{R}}$ with non-negligible probability that $\mathbf{E}_7$ happens. When $\mathbf{E}_7$ happens, the $1 - \alpha$ fraction of honest miners would send (pre_s, pre_a) to $C^{\mathsf{A}}_{\mathsf{burn}}$, if they are chosen to mine a block. By Lemma 5.5, if $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\text{Ƀ}x_b - \text{Å}x_a - \text{Å}c^{\mathsf{A}}_a) + \$E$. On the other hand, if $C^{\mathsf{A}}_{\mathsf{burn}}$ is not activated by an honest miner, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\text{Ƀ}x_b) + \$E$. When $\mathbf{E}_7$ happens, the probability

that $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner is at least $1-\alpha$. Thus, by the same calculation as the previous case, the utility of $\mathcal{A}$ is strictly less than $\$\mathsf{AV}(-\ddot{\mathbb{A}}x_a)$.

Event $\mathbf{E}_8$ happens first. Consider some strategy $S_8 \in \overline{\mathcal{R}}$ with non-negligible probability that $\mathbf{E}_8$ happens. If Bob enters the execution phase, because Alice does not send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before $\mathsf{BobChain}$ time T^{B}_1, Bob will send pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$ as soon as $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated. On the other hand, if Bob enters the abort phase, because Alice does not send ping to $P^{\mathsf{B}}_{\mathsf{refund}}$ before $\mathsf{AliceChain}$ time T^{A}_1, Bob will send pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$ as soon as $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated. In either case, Bob always sends pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$ as soon as $A^{\mathsf{A}}_{\mathsf{defuse}}$ is activated.

If $A^{\mathsf{A}}_{\mathsf{defuse}}$ is guaranteed to be activated at $\mathsf{AliceChain}$ time t^*, it must be activated no later than $\mathsf{AliceChain}$ time t^*+1. When $\mathbf{E}_8$ happens, $A^{\mathsf{A}}_{\mathsf{defuse}}$ has been guaranteed to be activated. Thus, when $\mathbf{E}_8$ happens, pre_a and pre_b are both publicly known. Then, the $1-\alpha$ fraction of honest miners would send (pre_a, pre_b) to $C^{\mathsf{A}}_{\mathsf{burn}}$ if they are chosen to mine a block. By Lemma 5.5, if $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b - \ddot{\mathbb{A}}x_a - \ddot{\mathbb{A}}c^{\mathsf{A}}_a) + \$E$. On the other hand, if $C^{\mathsf{A}}_{\mathsf{burn}}$ is not activated by an honest miner, the utility of $\mathcal{A}$ is at most $\$\mathsf{AV}(\ddot{\mathbb{B}}x_b) + \$E$. When $\mathbf{E}_8$ happens, the probability that $C^{\mathsf{A}}_{\mathsf{burn}}$ is activated by an honest miner is at least $1-\alpha$. Thus, by the same calculation as the previous case, the utility of $\mathcal{A}$ is strictly less than $\$\mathsf{AV}(-\ddot{\mathbb{A}}x_a)$.

Finally, notice that the above analysis holds even if $\mathcal{A}$ may post a new contract on the fly during the protocol execution, since all other players are honest and will not deposit money into the new contract.

□

Lemma 5.11 (Against Externally Incentivized Alice-Miner Coalition)**.** *Suppose that the hash function $H(\cdot)$ is a one-way function. Let $\mathcal{C}$ be a coalition consisting of Bob and possibly any subset of the miners, and let $\mathcal{A}$ be a disjoint coalition consisting of Alice and at most α fraction of the mining power where $\alpha \in [0, 1-1/\mathsf{poly}(\lambda)]$. Suppose that $\mathcal{C}$ does not have external incentives but $\mathcal{A}$ may have up to $\$E$ amount of external incentives. Let $S_{\mathcal{A}}$ be an arbitrary PPT strategy of $\mathcal{A}$ that is not in $\overline{\mathcal{R}}$. Then, there exists a negligible function $\mathsf{negl}(\cdot)$ such that except with negligible probability $\mathsf{negl}(\lambda)$, it holds that*

$$\mathsf{util}^{\mathcal{C}}(HS_{\mathcal{C}}, S_{\mathcal{A}}, HS_{\mathcal{D}}) \geq 0,$$

where $\mathcal{D}$ denotes everyone else not in $\mathcal{C} \cup \mathcal{A}$.

Proof. By Lemma 5.7, any strategy that makes one of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7, \mathbf{E}_8$ happen is blatantly irrational. By Lemma 5.9, if none of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7, \mathbf{E}_8$ happen, the one of $(\mathsf{Normal}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}})$, $(\mathsf{Refund}^{\mathsf{B}} + \mathsf{Normal}^{\mathsf{A}})$, and $(\mathsf{Refund}^{\mathsf{B}} + \mathsf{Refund}^{\mathsf{A}})$ must happen. According to Table 3, because $\$\mathsf{BV}(-\ddot{\mathbb{B}}x_b + \ddot{\mathbb{A}}x_a) > 0$, for all three possible cases, Bob's utility is never negative. □

5.2.4 Combine Everything Together

Now, we are ready to prove that the atomic swap protocol satisfies bounded maximin fairness.

Theorem 5.12 (Bounded maximin fairness)**.** *Suppose that $H(\cdot)$ is a one-way function and suppose the parameters $\ddot{\mathbb{B}}c^{\mathsf{B}}_a, \ddot{\mathbb{A}}c^{\mathsf{A}}_a, \ddot{\mathbb{B}}c^{\mathsf{B}}_b, \ddot{\mathbb{A}}c^{\mathsf{A}}_b, \alpha \in [0, 1-1/\mathsf{poly}(\lambda)]$ satisfy the constraints in Figure 2. Then, the atomic swap protocol satisfies α-bounded maximin fairness against external incentives.*

Proof. Let $\mathcal{C}$ be a set of honest players where we want to show that $\mathcal{C}$'s utility is non-negative, and let $\mathcal{C}'$ be an externally incentivized coalition. There are five cases to consider.

Case 1: Alice $\in \mathcal{C}$ and Bob $\in \mathcal{C}'$: covered by Lemma 5.8.

Case 2: Bob $\in \mathcal{C}$ and Alice $\in \mathcal{C}'$: covered by Lemma 5.11.

Case 3: $\mathcal{C}$ is miner-only: It is straightforward to see that no matter how players outside $\mathcal{C}$ behave, as long as $\mathcal{C}$ behaves honestly, its utility is non-negative.

Case 4: Alice $\in \mathcal{C}$ and $\mathcal{C}'$ is miner-only: Notice that both Alice and Bob are assumed to be honest. In the protocol, Alice and Bob decide whether they will go to the abort phase according to whether Bob sends pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$. Thus, when Alice and Bob are both honest, both of them enter the execution phase or both of them enter the abort phase. There are two subcases.

- *Subcase 1: Both Alice and Bob enter the execution phase.* Bob only sends pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$ when $B^{\mathsf{B}}_{\mathsf{defuse}}$ has been activated. Ignoring the negligible probability that $\mathcal{C}'$ finds pre_c by itself, $B^{\mathsf{B}}_{\mathsf{burn}}$ can never be activated. Alice would send pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ as soon as she enters the execution phase, and Bob would send pre_s to $P^{\mathsf{A}}_{\mathsf{default}}$ as soon as Alice sent pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$. Because Alice always sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before $\mathsf{BobChain}$ time T^{B}_1, Bob never sends pre_b. Besides, Alice never sends pre_a. Ignoring the negligible probability that $\mathcal{C}'$ finds pre_b or pre_a by itself, $A^{\mathsf{A}}_{\mathsf{burn}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$ and $C^{\mathsf{A}}_{\mathsf{burn}}$ can never be activated. When Alice and Bob are in the execution phase, $P^{\mathsf{B}}_{\mathsf{refund}}$ can be activated only by Bob sending pre_b. As we have shown, Bob never sends pre_b, so $P^{\mathsf{B}}_{\mathsf{refund}}$ can never be activated. Moreover, when Alice and Bob are in the execution phase, $P^{\mathsf{A}}_{\mathsf{refund}}$ can be activated only by Bob sending ping. However, Bob will only send ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ if $C^{\mathsf{B}}_{\mathsf{refund}}$ is activated. Because $P^{\mathsf{B}}_{\mathsf{refund}}$ cannot be activated, $C^{\mathsf{B}}_{\mathsf{refund}}$ cannot be activated either. Thus, $P^{\mathsf{A}}_{\mathsf{refund}}$ cannot ever be activated. The $1-\alpha$ fraction of honest miner will include Alice's and Bob's transactions, so except the negligible probability, $P^{\mathsf{B}}_{\mathsf{default}}$ and $P^{\mathsf{A}}_{\mathsf{default}}$ will be activated in polynomial time. Thus, the honest $\mathcal{C}$ obtains non-negative utility.
- *Subcase 2: Both Alice and Bob go to the abort phase.* In this case, Bob never sends pre_c. In the abort phase Alice always sends ping to $P^{\mathsf{B}}_{\mathsf{refund}}$ at $\mathsf{AliceChain}$ time T^{A}_0, so Bob never sends pre_b. Similarly, Bob always sends ping to $P^{\mathsf{A}}_{\mathsf{refund}}$ at $\mathsf{AliceChain}$ time T^{A}_0, so Alice never sends pre_a. Moreover, Alice never sends pre_s. Ignoring the negligible probability that $\mathcal{C}'$ finds $pre_s, pre_a, pre_b, pre_c$ by itself, $A^{\mathsf{A}}_{\mathsf{burn}}$, $B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{default}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{default}}$, $C^{\mathsf{A}}_{\mathsf{burn}}$ can never be activated.

  If $\textsc{Contract}^{\mathrm{B}}$ ($\textsc{Contract}^{\mathrm{A}}$, resp.) has not been active, Alice and Bob send the withdrawal transaction to $\textsc{Contract}^{\mathrm{B}}$ ($\textsc{Contract}^{\mathrm{A}}$, resp.). The $1-\alpha$ fraction of honest miner will include Alice's and Bob's withdrawal transactions, so except the negligible probability, they can get their deposit back unless the contract becomes active. Next, we analyze different cases depending on which contract becomes active.

  - $\textsc{Contract}^{\mathrm{B}}$ *is active.* As we have shown, $B^{\mathsf{B}}_{\mathsf{burn}}$, $P^{\mathsf{B}}_{\mathsf{default}}$, $C^{\mathsf{B}}_{\mathsf{burn}}$ cannot be activated. The $1-\alpha$ fraction of honest miner will include Alice's transaction, ping to $P^{\mathsf{B}}_{\mathsf{refund}}$, once they mine a block. Thus, $P^{\mathsf{B}}_{\mathsf{refund}}$ will be activated in polynomial time. When τ^{B} $\mathsf{BobChain}$ time has passed since $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, Alice and Bob will send ping to $C^{\mathsf{B}}_{\mathsf{refund}}$. Again, the $1-\alpha$ fraction of honest miner will include Alice's and Bob's transactions, ping to $C^{\mathsf{B}}_{\mathsf{refund}}$, once they mine a block. Thus, $C^{\mathsf{B}}_{\mathsf{refund}}$ will be activated in polynomial time.
  - $\textsc{Contract}^{\mathrm{A}}$ *is active.* As we have shown, $A^{\mathsf{A}}_{\mathsf{burn}}$, $P^{\mathsf{A}}_{\mathsf{default}}$, $C^{\mathsf{A}}_{\mathsf{burn}}$ cannot be activated. The $1-\alpha$ fraction of honest miner will include Bob's transaction, ping to $P^{\mathsf{A}}_{\mathsf{refund}}$, once they mine a block. Thus, $P^{\mathsf{A}}_{\mathsf{refund}}$ will be activated in polynomial time. When τ^{A} $\mathsf{AliceChain}$ time has passed since $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated, Alice and Bob will send ping to $C^{\mathsf{A}}_{\mathsf{refund}}$. Again, the $1-\alpha$ fraction of honest miner will include Alice's and Bob's transactions, ping to $C^{\mathsf{A}}_{\mathsf{refund}}$, once they mine a block. Thus, $C^{\mathsf{A}}_{\mathsf{refund}}$ will be activated in polynomial time.

In all cases, $\mathcal{C}$'s utility is non-negative except with negligible probability.

Case 5: Bob $\in \mathcal{C}$ and $\mathcal{C}'$ is miner-only: The argument is the same as Case 4. □

5.3 Achieving Dropout Resilience

Theorem 5.13 (Dropout resilience of atomic swap)**.** *Suppose that $H(\cdot)$ is a one-way function and that all players are PPT machines. Our atomic swap protocol is dropout resilient. In other words, suppose at least $1/\mathsf{poly}(\lambda)$ fraction of the mining power is honest on either chain; if either Alice or Bob plays honestly but drops out before the end of the protocol, then with $1-\mathsf{negl}(\lambda)$ probability, the other party's utility must be non-negative.*

Proof. Throughout the proof, for any $X \in \{pre_s, pre_a, pre_b, pre_c\}$, we ignore the negligible probability that the miners can find the preimage X by itself if Alice and Bob have never sent X before.

We first analyze the cases where Alice drops out. If any of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7, \mathbf{E}_8$ is about to happen, it must be that Alice deviates from the protocol and is about to send a transaction that does not follow from the honest protocol. Because Alice is honest, no matter when Alice drops out, none of $\mathbf{E}_5, \mathbf{E}_6, \mathbf{E}_7, \mathbf{E}_8$ would happen. By Lemma 5.9, one and only one of ($\mathsf{Normal}^{\mathsf{B}}+\mathsf{Normal}^{\mathsf{A}}$), ($\mathsf{Refund}^{\mathsf{B}}+\mathsf{Normal}^{\mathsf{A}}$) and ($\mathsf{Refund}^{\mathsf{B}}+\mathsf{Refund}^{\mathsf{A}}$) will happen. According to Table 3, because $\$\mathsf{BV}(\text{Å}x_a - \text{Ƀ}x_b) > 0$, for any of the three cases, Bob's utility is non-negative.

Next, we analyze the cases where Bob drops out. If any of $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3, \mathbf{E}_4$ is about to happen, it must be that Bob deviates from the protocol and is about to send a transaction that does not follow from the honest protocol. Because Bob is honest, no matter when Bob drops out, none of $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3, \mathbf{E}_4$ would happen. By Lemma 5.6, one and only one of ($\mathsf{Normal}^{\mathsf{B}}+\mathsf{Normal}^{\mathsf{A}}$), ($\mathsf{Normal}^{\mathsf{B}}+\mathsf{Refund}^{\mathsf{A}}$) and ($\mathsf{Refund}^{\mathsf{B}}+\mathsf{Refund}^{\mathsf{A}}$) will happen. According to Table 4, because $\$\mathsf{AV}(\text{Ƀ}x_b - \text{Å}x_a) > 0$, for any of the three cases, Alice's utility is non-negative. □

6 Rational Defection from Grand Coalition

In Theorem 4.1, we prove that the atomic swap protocol is secure assuming at least $1/\mathsf{poly}(\lambda)$ fraction of honest miners. Here, we justify this assumption by considering a metagame that captures the formation of the coalition. In more detail, we argue that when everyone else joins the coalition, a rational miner will prefer to defect the coalition. In other words, a grand coalition consisting 100% of the mining power is unstable if each individual miner is rational.

6.1 Disincentivizing the Grand Coalition Absent of External Incentive.

We first assume the coalition is formed without any external incentive, and we show that any grand coalition whose strategic utility is higher than the honest utility is not stable, and thus justify the assumption of $1/\mathsf{poly}(\lambda)$ fraction of honest miners for CSP-fairness. The analysis is similar to the metagame for knowledge-coin exchange in [CMST22b].

We start with Alice-miner coalition. According to Lemma 5.2, their utility is better than the honest case only if $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{A}}$ both happen, which leads to a strategic utility $\text{Ƀ}x_b$. Suppose Alice controls 100% of the mining power, and they adopt a strategy that leads to $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{A}}$ with non-negligible probability. When $\mathsf{Normal}^{\mathsf{B}}$ happens, i.e., $P^{\mathsf{B}}_{\mathsf{default}}$ is activated, Bob must have sent pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$, and it implies $\textsc{Contract}^{\mathrm{A}}$ has been active and Bob enters the execution phase. If $\textsc{Contract}^{\mathrm{A}}$ is active, $(P^{\mathsf{A}}_{\mathsf{refund}} + C^{\mathsf{A}}_{\mathsf{refund}})$ is the only possibility for $\mathsf{Refund}^{\mathsf{A}}$. However, in the execution phase, Bob sends ping to $P^{\mathsf{A}}_{\mathsf{default}}$ only after $P^{\mathsf{B}}_{\mathsf{default}}$ has been activated. Thus, the fact that $\mathsf{Normal}^{\mathsf{B}}$ and $\mathsf{Refund}^{\mathsf{A}}$ both happen implies that $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated by pre_a. When Bob enters the execution phase, there are two possible cases:

- *Alice sends pre_s to $P^{\mathsf{B}}_{\mathsf{default}}$ before T^{B}_1.* Because $P^{\mathsf{A}}_{\mathsf{refund}}$ can only be activated after $T^{\mathsf{A}}_1 > T^{\mathsf{B}}_1$, the moment that $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated, both pre_s and pre_a are publicly known. Within the time window τ^{A} that $C^{\mathsf{A}}_{\mathsf{refund}}$ can be activated, anyone can send $(pre_s + pre_a)$ to $C^{\mathsf{A}}_{\mathsf{burn}}$.
- *Alice does not send pre_s before T^{B}_1.* In this case, either $A^{\mathsf{A}}_{\mathsf{defuse}}$ has not been activated, or Bob has sent pre_b to $P^{\mathsf{B}}_{\mathsf{refund}}$. If $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated, anyone can either send pre_a to $A^{\mathsf{A}}_{\mathsf{burn}}$ or send $(pre_a + pre_b)$ to $C^{\mathsf{A}}_{\mathsf{burn}}$.

Recall that the coalition's strategic utility is at most $\text{฿}x_b$. Suppose the coalition distributes the strategic gain proportionally to the mining power, and consider a small miner i that has a small mining power γ. If i joins the coalition and cooperates, its expected reward is at most $p\gamma \cdot \text{฿}x_b$, where p denotes the probability that $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated. On the other hand, suppose i chooses to not join the coalition. Since its influence to the block generation process is small, we may assume that $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated with probability p or more. Without loss of generality, we may assume that every miner in the coalition commits to starving $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ in every block they mine, e.g., by placing a collateral that it will honor its commitment — if not, then the coalition will not be stable since a coalition member will be incentivized to defect from the coalition and claim $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$ itself, which is what we want to prove.

As we have shown above, since the moment $P^{\mathsf{A}}_{\mathsf{refund}}$ is activated, miner i has a τ^{A} lead in time to mine a block in which i can redeem $\$\epsilon$ from $A^{\mathsf{A}}_{\mathsf{burn}}$ or $C^{\mathsf{A}}_{\mathsf{burn}}$. The probability that i mines a block in a window of τ^{A} blocks is $1-(1-\gamma)^{\tau^{\mathsf{A}}}$. Therefore, if i does not join the coalition, its expected gain would be at least $p \cdot \text{฿}\epsilon \cdot 1-(1-\gamma)^{\tau^{\mathsf{A}}}$. If i joins the coalition, its expected gain is $p\gamma \cdot \text{฿}x_b$. Thus, as long as $p \cdot \text{฿}\epsilon \cdot (1-(1-\gamma)^{\tau^{\mathsf{A}}}) > p\gamma \cdot \text{฿}x_b$, i's best strategy is to not join the coalition. This means that if everyone else joins the coalition, some small miner i wants to defect.

Next, we analyze Bob-miner coalition. By Lemma 5.2, the coalition's utility is better off only if $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ both happen. $\mathsf{Normal}^{\mathsf{A}}$ happens, i.e., $P^{\mathsf{A}}_{\mathsf{default}}$ is activated, only if Bob sends pre_s or Alice sends ping. However, Alice sends ping to $P^{\mathsf{A}}_{\mathsf{default}}$ only if $P^{\mathsf{B}}_{\mathsf{default}}$ has been activated which leads to $\mathsf{Normal}^{\mathsf{B}}$. Thus, we assume that $P^{\mathsf{A}}_{\mathsf{default}}$ is activated by pre_s. Ignoring the negligible probability that Bob finds pre_s by itself, Alice only sends pre_s if Bob has sent pre_c to $P^{\mathsf{B}}_{\mathsf{default}}$ and $\textsc{Contract}^{\mathrm{B}}$ is active. Thus, the fact that $\mathsf{Refund}^{\mathsf{B}}$ happens implies that $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated by pre_b. Consequently, whenever $P^{\mathsf{B}}_{\mathsf{refund}}$ is activated, pre_s, pre_b, and pre_c are all publicly known, and anyone can send all of them to $C^{\mathsf{B}}_{\mathsf{burn}}$. By the similar argument as Alice-miner coalition, if everyone else joins the coalition, some small miner i controlling γ fraction of the mining power wants to defect.

6.2 Disincentivizing the Grand Coalition in the Presence of External Incentive.

Now we consider the strategic coalition with external incentive, and we show that any grand coalition that harms the honest players is not stable, and thus justify the assumption of $1/\mathsf{poly}(\lambda)$ fraction of honest miners for bounded maximin fairness.

Let $\$E$ be the upper bound of the external incentive, and let $\$V$ be the maximum value of the atomic swap protocol to the coalition, where $\$V \leq \$E + \text{฿}x_b$ to Alice-miner coalition and $\$V \leq \$E + \text{Ł}x_a$ to Bob-miner coalition. We start with Alice-miner coalition. According to Table 3, the honest Bob's utility is negative if any of the following conditions hold: 1) $\mathsf{Refund}^{\mathsf{A}}$ and $\mathsf{Normal}^{\mathsf{B}}$ both happen; 2) $\mathsf{Burn}^{\mathsf{A}}$ happens; 3) $\mathsf{Burn}^{\mathsf{B}}$ happens.

By the choice of $\text{Ł}c^{\mathsf{A}}_a$ and $\text{฿}c^{\mathsf{B}}_a$ (Figure 2), once $\mathsf{Burn}^{\mathsf{A}}$ or $\mathsf{Burn}^{\mathsf{B}}$ happens, the coalition's utility is negative even if it is compensated by the external incentive. Thus, the only profitable strategy for the externally incentivized coalition that may harm honest players is to invoke $\mathsf{Refund}^{\mathsf{A}}$ and $\mathsf{Normal}^{\mathsf{B}}$.

In this case, we can follow the same argument as in Section 6.1. As long as $p \cdot \$\epsilon \cdot (1-(1-\gamma)^{\tau^{\mathsf{A}}}) > p\gamma \cdot \$V$, the small miner i's best strategy is to not join the coalition.

Similarly, we can show that Bob-miner coalition is not stable. According to Table 4, the honest Alice's utility is negative if any of the following conditions hold: 1) $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$ both happen; 2) $\mathsf{Burn}^{\mathsf{A}}$ happens; 3) $\mathsf{Burn}^{\mathsf{B}}$ happens. By the choice of $₳c_b^{\mathsf{A}}$ and $₿c_b^{\mathsf{B}}$, once $\mathsf{Burn}^{\mathsf{A}}$ or $\mathsf{Burn}^{\mathsf{B}}$ happens, the coalition's utility is negative even if it is compensated by the external incentive. Thus, the only profitable strategy for the externally incentivized coalition that may harm honest players is to invoke $\mathsf{Refund}^{\mathsf{B}}$ and $\mathsf{Normal}^{\mathsf{A}}$. In this case, we can follow the same argument as in Section 6.1, as long as $p \cdot \$\epsilon \cdot (1-(1-\gamma)^{\tau^{\mathsf{A}}}) > p\gamma \cdot \$V$, the small miner i's best strategy is to not join the coalition.

7 Instantiation

We now discuss the instantiation of our scheme. We first implement it using a general-purpose smart contract language, and then show an instantiation in Bitcoin's UTXO model.

7.1 Ethereum Instantiation

We implemented our contracts in 340 LoC in Solidity, Ethereum's smart contract language. Transaction fees on Ethereum are determined by gas usage, which corresponds to the total cost of operations executed by the contract.

We give the gas cost of our scheme in Table 5. Additionally, we compare our gas cost to those of Rapidash's CSP-fair swap [CMST22b] in Table 6. As expected, our costs are slightly higher overall due to the extra input logic and the extra defuse logic that are necessary to achieve bounded maximin fairness. However, we note that once the contracts are active, the optimistic paths have similar costs in both schemes.

Table 5: Bounded maximin fair atomic swap, gas cost. (O) denotes an optimistic case where both Alice and Bob are honest.

Contract	**Redeem path**	**Gas**
$\textsc{Contract}^{\mathrm{B}}$	Input, Alice	48,355
	Input, Bob	50,583
	Withdraw, Alice	35,956
	Withdraw, Bob	38,271
	(O) ($P^{\mathsf{B}}_{\mathsf{default}}$), Alice	35,405
	(O) ($B^{\mathsf{B}}_{\mathsf{defuse}} + P^{\mathsf{B}}_{\mathsf{default}}$), Bob	88,399
	Refund ($P^{\mathsf{B}}_{\mathsf{refund}} + C^{\mathsf{B}}_{\mathsf{refund}}$), Alice	114,662
	Refund ($B^{\mathsf{B}}_{\mathsf{defuse}} + P^{\mathsf{B}}_{\mathsf{refund}} + C^{\mathsf{B}}_{\mathsf{refund}}$), Bob	147,567
	Early Bomb ($B^{\mathsf{B}}_{\mathsf{burn}}$), Miner	50,295
	Bomb ($C^{\mathsf{B}}_{\mathsf{burn}}$), Miner	57,152
$\textsc{Contract}^{\mathrm{A}}$	Input, Alice	50,650
	Input, Bob	48,333
	Withdraw, Alice	38,251
	Withdraw, Bob	35,912
	(O) ($P^{\mathsf{A}}_{\mathsf{default}}$), Alice	54,925
	(O)($P^{\mathsf{A}}_{\mathsf{default}}$), Bob	58,679
	Refund ($A^{\mathsf{A}}_{\mathsf{defuse}} + P^{\mathsf{A}}_{\mathsf{refund}} + C^{\mathsf{A}}_{\mathsf{refund}}$), Alice	149,634
	Refund ($A^{\mathsf{A}}_{\mathsf{defuse}} + P^{\mathsf{A}}_{\mathsf{refund}} + C^{\mathsf{A}}_{\mathsf{refund}}$), Bob	145,894
	Early bomb ($A^{\mathsf{A}}_{\mathsf{burn}}$), Miner	49,956
	Bomb ($C^{\mathsf{A}}_{\mathsf{burn}}$), Miner	53,475

Table 6: Gas cost comparison between Rapidash and our bounded maximin fair atomic swap. (O) denotes an optimistic case.

Contract	Redeem path	Gas
Rapidash^{B}	Normal path ($P^{B}_{default}$), Alice	52,279
	Normal path ($P^{B}_{default}$), Bob	56,681
	Refund path ($P^{B}_{refund} + C^{B}_{refund}$), Bob	123,631
	Burn path (C^{B}_{burn}), Miner	42,266
Rapidash^{A}	Input, Alice	50,465
	Input, Bob	55,817
	Withdraw, Alice	38,228
	Withdraw, Bob	35,911
	(O) ($P^{A}_{default}$), Alice	54,904
	(O) ($P^{A}_{default}$), Bob	58,656
	Refund ($P^{A}_{refund} + C^{A}_{refund}$), Alice	118,379
	Refund ($P^{A}_{refund} + C^{A}_{refund}$), Bob	114,647
	Burn (C^{A}_{burn}), Miner	53,431
Contract^{B}	Input, Alice	48,355
	Input, Bob	50,583
	Withdraw, Alice	35,956
	Withdraw, Bob	38,271
	(O) ($P^{B}_{default}$), Alice	35,405
	(O) ($B^{B}_{defuse} + P^{B}_{default}$), Bob	88,399
	Refund ($P^{B}_{refund} + C^{B}_{refund}$), Alice	114,662
	Refund ($B^{B}_{defuse} + P^{B}_{refund} + C^{B}_{refund}$), Bob	147,567
	Early Bomb (B^{B}_{burn}), Miner	50,295
	Bomb (C^{B}_{burn}), Miner	57,152
Contract^{A}	Input, Alice	50,650
	Input, Bob	48,333
	Withdraw, Alice	38,251
	Withdraw, Bob	35,912
	(O) ($P^{A}_{default}$), Alice	54,925
	(O)($P^{A}_{default}$), Bob	58,679
	Refund ($A^{A}_{defuse} + P^{A}_{refund} + C^{A}_{refund}$), Alice	149,634
	Refund ($A^{A}_{defuse} + P^{A}_{refund} + C^{A}_{refund}$), Bob	145,894
	Early bomb (A^{A}_{burn}), Miner	49,956
	Bomb (C^{A}_{burn}), Miner	53,475

7.2 Bitcoin Instantiation

Traditional smart contracts operate by receiving coins from users, holding them within the contract until specific conditions activate the transfer. Once triggered, the contract disperses some or all of the coins to designated recipients. Bitcoin, however, follows the Unspent Transaction Output (UTXO) model, where coins are linked to addresses represented as $Adr \in \{0,1\}^\lambda$. Each address can be spent only once, ensuring that once used in a transaction, it cannot be reused. Transactions are recorded on the blockchain, facilitating the movement of coins from input addresses to new output addresses. Any excess coins not explicitly allocated are collected by the miner as a transaction fee for processing the block.

Bitcoin transactions can be thought of as being generated by the transaction function tx. A transaction tx_A, denoted

$$tx_A := tx\begin{pmatrix}[(Adr_1, \Phi_1, \$v_1), \ldots, (Adr_n, \Phi_n, \$v_n)],\\ [(Adr'_1, \Phi'_1, \$v'_1), \ldots, (Adr'_m, \Phi'_m, \$v'_m)]\end{pmatrix},$$

transfers v_i coins from each input address Adr_i, $i \in [n]$, and deposits v'_i coins to each output address Adr'_j, $j \in [m]$. To prevent minting coins from thin air, it must be guaranteed that $\sum_{i\in[n]} \$v_i \geq \sum_{j\in[m]} \$v'_j$, where the difference is offered as transaction fee to the miners.

Bitcoin addresses are governed by *scripts* $\Phi : \{0,1\}^\lambda \to \{0,1\}$, which define spending conditions for their coins. Unlike smart contracts, which allow arbitrary logic, Bitcoin's scripting language has limited expressiveness. A transaction is *authorized* when witnesses $[x_1, \ldots, x_n]$ satisfy $\Phi_i(x_i) = 1$ for all i, and *confirmed* once recorded on the blockchain.

Since addresses can only be spent *once*, contract logic must be encoded directly within scripts. Our approach relies on standard Bitcoin scripts, ensuring compatibility. An address Adr is controlled by a user if they possess a valid witness x such that $\Phi(x) = 1$.

7.2.1 Instantiating Contract$^{\mathsf{B}}$ with Bounded Maximin Fairness

We depict the payment flow in Figure 7. Alice and Bob use transaction $tx^{\mathsf{B}}_{\mathsf{stp}}$ that creates $Adr^{\mathsf{B}}_{\mathsf{stp}}$ with $\$x_b + \$c^B_a + \$c^B_b$ coins and another address $Adr_{B^{\mathsf{B}}_{\mathsf{defuse}}}$ with $\$\eta^B$ coins in it. Here $x_b + \$c^B_b + \$\eta^B$ coins come from Bob and $\$c^B_a$ coins come from Alice. We have two new transactions $tx_{B^{\mathsf{B}}_{\mathsf{defuse}}}$ and $tx_{B^{\mathsf{B}}_{\mathsf{burn}}}$, corresponding to the activation points $B^{\mathsf{B}}_{\mathsf{defuse}}$ and $B^{\mathsf{B}}_{\mathsf{burn}}$, respectively. Transaction $tx_{B^{\mathsf{B}}_{\mathsf{defuse}}}$ redeems $\$\eta^{\mathrm{B}}$ coins from the address $Adr_{B^{\mathsf{B}}_{\mathsf{defuse}}}$ provided a timeout of T^{B}_1 has passed since the setup transaction was published on the blockchain. The other transaction $tx_{B^{\mathsf{B}}_{\mathsf{burn}}}$ redeems the $\$\eta^{\mathrm{B}}$ coins from $Adr_{B^{\mathsf{B}}_{\mathsf{defuse}}}$, as well as the coins from the setup address $Adr^{\mathrm{B}}_{\mathsf{stp}}$ provided pre_c is released. The transaction burns $(\$x_b + \$c^{\mathsf{B}}_a + \$c^{\mathsf{B}}_b + \$\eta^{\mathrm{B}} - \$\epsilon^{\mathsf{B}})$ coins leaving behind $\$\epsilon^{\mathsf{B}}$ coins as transaction fees.

Transaction $tx_{P^{\mathsf{B}}_{\mathsf{default}}}$ can only be activated with pre_s (pre-image of h_s) and z_c (pre-image of h_c) are available. This transaction spends coins from Adr^B_{stp}, and sends $\$x_b + \$c^B_a$ coins to an address of Alice and send $\$c^B_b$ coins to an address of Bob. We then have two cases: two transactions $tx_{P^{\mathsf{B}}_{\mathsf{refund}}}$ and $tx^{\mathsf{ping}}_{P^{\mathsf{B}}_{\mathsf{refund}}}$ can activate $P^{\mathsf{B}}_{\mathsf{refund}}$ and spend from Adr^B_{stp} after a timeout of T^B_1 has passed. The only difference is $tx_{P^{\mathsf{B}}_{\mathsf{refund}}}$ can be activated by Bob only with pre-image pre_b, while anyone, especially Alice can publish $tx^{\mathsf{ping}}_{P^{\mathsf{B}}_{\mathsf{refund}}}$ without needing any pre-image. If either one of those transactions are published, after a timeout of τ^B, Alice and Bob can get a full refund of their locked coins with transactions $tx_{C^{\mathsf{B}}_{\mathsf{refund}}}$ and $tx^{\mathsf{ping}}_{C^{\mathsf{B}}_{\mathsf{refund}}}$, respectively. However, before the timeout, the $C^{\mathsf{B}}_{\mathsf{burn}}$ path can be activated in both cases before the timeout τ^B passes using $tx^{P^{\mathsf{B}}_{\mathsf{refund}}}_{C^{\mathsf{B}}_{\mathsf{burn}}}$ and $tx^{P^{\mathsf{B}}_{\mathsf{refund}},\mathsf{ping}}_{C^{\mathsf{B}}_{\mathsf{burn}}}$, respectively. In

both cases, all of pre_s, pre_b, pre_c must be available. Finally we have transaction $tx_{C^{\mathsf{B}}_{\mathsf{burn}}}$ that can also directly activate $C^{\mathsf{B}}_{\mathsf{burn}}$ path from Adr^{B}_{stp} given all of pre_s, pre_b, pre_c. All transactions must be signed by both Alice and Bob.

Protocol Flow. Alice and bob first agree on the setup transaction $tx^{\mathrm{B}}_{\mathsf{stp}}$ and sign all other redeeming transactions. They ensure that the signatures on the transaction $tx_{B^{\mathsf{B}}_{\mathsf{defuse}}}$ are only with Bob, and the signatures on $tx^{\mathsf{ping}}_{P^{\mathsf{B}}_{\mathsf{refund}}}$ are only with Alice. Alice and Bob keep these signatures privately and broadcast all other transactions and signatures into the network. By posting the setup transaction on the blockchain, we enter the execution phase.

Whenever Bob wishes to activate $B^{\mathsf{B}}_{\mathsf{defuse}}$ branch, he publishes $tx_{B^{\mathsf{B}}_{\mathsf{defuse}}}$ along with the corresponding signatures on the blockchain. Notice that since $tx_{B^{\mathsf{B}}_{\mathsf{defuse}}}$ and $tx_{B^{\mathsf{B}}_{\mathsf{burn}}}$ spend from the address $Adr_{B^{\mathsf{B}}_{\mathsf{defuse}}}$, they are mutually exclusive, meaning only one of them can be posted on the blockchain. Notice that any of the other branches can be activated if and only if $B^{\mathsf{B}}_{\mathsf{defuse}}$ was activated (or in other words, $B^{\mathsf{B}}_{\mathsf{burn}}$ was not activated with $tx_{B^{\mathsf{B}}_{\mathsf{burn}}}$). Alice can publish $tx_{P^{\mathsf{B}}_{\mathsf{default}}}$ with pre-images pre_s and pre_c. If not, Bob can initiate the refund using $tx_{P^{\mathsf{B}}_{\mathsf{refund}}}$ after a timeout of T^{B}_1 and using the pre-image pre_b. Whenever Alice wants to activate $P^{\mathsf{B}}_{\mathsf{refund}}$ with a ping, she publishes the transaction $tx^{\mathsf{ping}}_{P^{\mathsf{B}}_{\mathsf{refund}}}$ along with valid signatures in her possession. If the transaction is published, activation point $C^{\mathsf{B}}_{\mathsf{refund}}$ can be activated by $tx^{\mathsf{ping}}_{C^{\mathsf{B}}_{\mathsf{refund}}}$ after a timeout of τ^{B} time. The $C^{\mathsf{B}}_{\mathsf{burn}}$ path can be activated using transactions $tx_{C^{\mathsf{B}}_{\mathsf{burn}}}$, $tx^{P^{\mathsf{B}}_{\mathsf{refund}}}_{C^{\mathsf{B}}_{\mathsf{burn}}}$ or $tx^{P^{\mathsf{B}}_{\mathsf{refund}},\mathsf{ping}}_{C^{\mathsf{B}}_{\mathsf{burn}}}$ depending on which paths have been activated so far.

7.2.2 Instantiating Contract$^{\mathsf{A}}$ with Bounded Maximin Fairness

The payment flow is depicted in Figure 8. Alice and Bob use transaction tx^{A}_{stp} that creates Adr^{A}_{stp} with $\$x_a + \$c^{A}_a + \$c^{A}_b$ coins and another address $Adr_{A^{\mathsf{A}}_{\mathsf{defuse}}}$ with $\$\eta^{A}$ coins in it. Here $x_a + \$c^{A}_a + \$\eta^{A}$ coins come from Alice and $\$c^{A}_b$ coins come from Bob. We have two new transactions $tx_{A^{\mathsf{A}}_{\mathsf{defuse}}}$ and $tx_{A^{\mathsf{A}}_{\mathsf{burn}}}$, corresponding to the activation points $A^{\mathsf{A}}_{\mathsf{defuse}}$ and $A^{\mathsf{A}}_{\mathsf{burn}}$, respectively. Transaction $tx_{A^{\mathsf{A}}_{\mathsf{defuse}}}$ redeems $\$\eta^{\mathrm{B}}$ coins from the address $Adr_{A^{\mathsf{A}}_{\mathsf{defuse}}}$ provided a timeout of T^{A}_1 has passed since the setup transaction was published on the blockchain. The other transaction $tx_{A^{\mathsf{A}}_{\mathsf{burn}}}$ redeems the $\$\eta^{\mathrm{A}}$ coins from $Adr_{A^{\mathsf{A}}_{\mathsf{defuse}}}$, as well as the coins from the setup address Adr^{A}_{stp} provided pre_a or pre_b is released. The transaction burns $(\$x_a + \$c^{\mathsf{A}}_a + \$c^{\mathsf{A}}_b + \$\eta^{\mathrm{A}} - \$\epsilon^{\mathsf{A}})$ coins leaving behind $\$\epsilon^{\mathsf{A}}$ coins as transaction fees.

We have transaction $tx^{\mathsf{ping}}_{P^{\mathsf{A}}_{\mathsf{default}}}$ that can activate $P^{\mathsf{A}}_{\mathsf{default}}$ by a ping by anyone. On the other hand, we have transaction $tx_{P^{\mathsf{A}}_{\mathsf{default}}}$ that can only be activated with pre_s (pre-image of h_s) is available. Both of these transactions will spend the coins from Adr^{A}_{stp}, and send $\$x_a + \$c^{A}_a$ coins to an address of Bob and send $\$c^{A}_b$ coins to an address of Alice. We then have two cases: two transactions $tx_{P^{\mathsf{A}}_{\mathsf{refund}}}$ and $tx^{\mathsf{ping}}_{P^{\mathsf{A}}_{\mathsf{refund}}}$ can activate $P^{\mathsf{A}}_{\mathsf{refund}}$ and spend from Adr^{A}_{stp} after a timeout of T^{A}_1 has passed. The only difference is $tx_{P^{\mathsf{A}}_{\mathsf{refund}}}$ can be activated by Alice only with pre-image pre_a, while anyone, especially Bob can publish $tx^{\mathsf{ping}}_{P^{\mathsf{A}}_{\mathsf{refund}}}$ without needing any pre-image. If either one of those transactions are published, after a timeout of τ^{B}, Alice and Bob can get a full refund of their locked coins with transactions $tx_{C^{\mathsf{A}}_{\mathsf{refund}}}$ and $tx^{\mathsf{ping}}_{C^{\mathsf{A}}_{\mathsf{refund}}}$, respectively. However, before the timeout, the $C^{\mathsf{A}}_{\mathsf{burn}}$ path can be activated in both cases using $tx^{P^{\mathsf{A}}_{\mathsf{refund}}}_{C^{\mathsf{A}}_{\mathsf{burn}}}$ and $tx^{P^{\mathsf{A}}_{\mathsf{refund}},\mathsf{ping}}_{C^{\mathsf{B}}_{\mathsf{burn}}}$, respectively. In both cases, either pre_a and

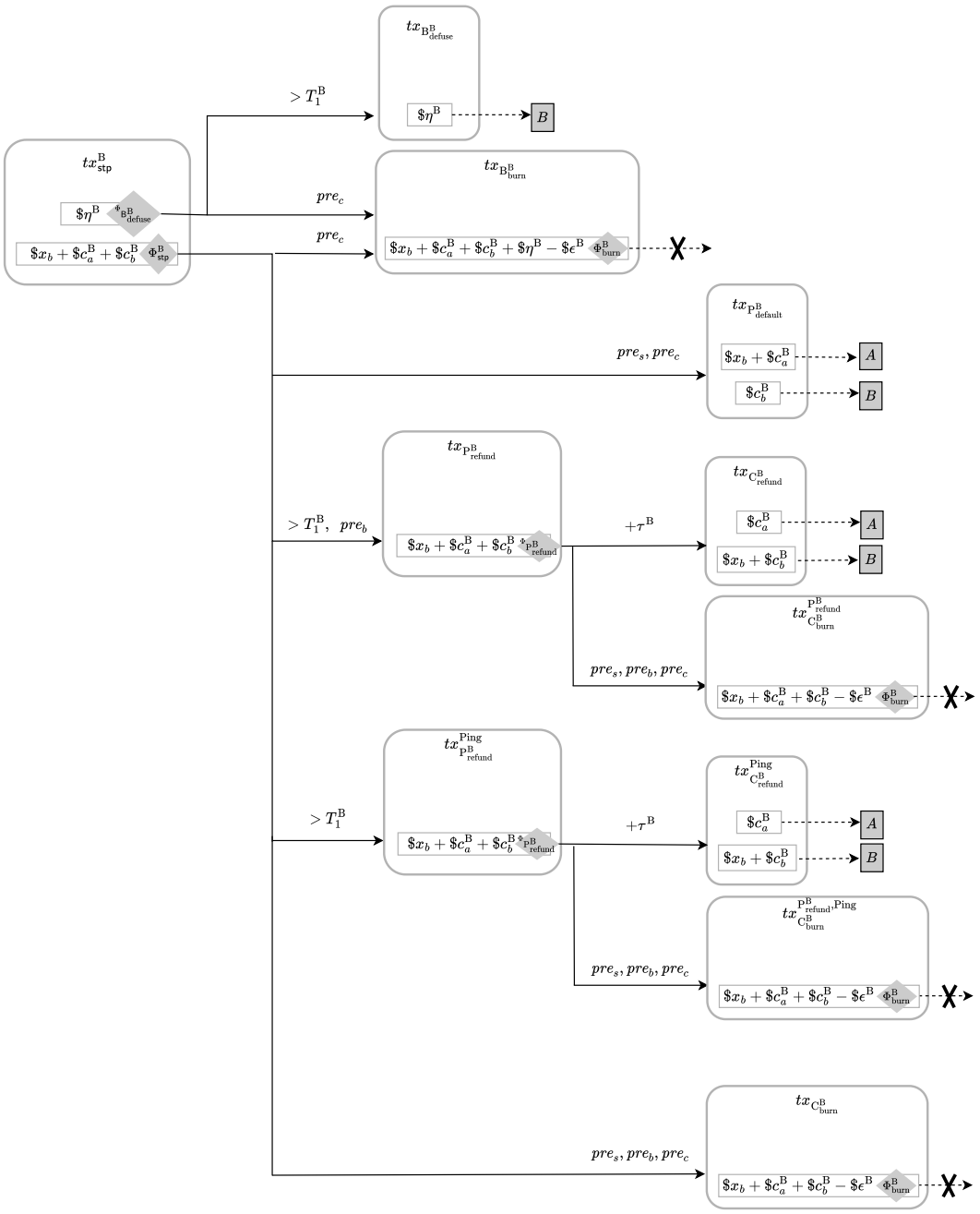

Figure 7: The transaction flow of $\textsc{Contract}^{\mathrm{B}}$ in Bitcoin for an atomic swap with bounded maximin fairness. Rounded boxes denote transactions, and rectangles within are outputs of the transaction. Incoming arrows denote transaction inputs, outgoing arrows denote how an output can be spent by a transaction at the end of the arrow. Solid lines indicate the transaction output can be spent only if both users sign the spending transaction. Dashed arrows indicate that the output can be spent by one user (A for Alice and B for Bob).

pre_s, or pre_a and pre_b must be available. Finally we have transaction $tx_{C'^{\mathsf{A}}_{\mathsf{burn}}}$ that can also directly activate $C'^{\mathsf{A}}_{\mathsf{burn}}$ path from Adr^{A}_{stp} given either pre_a and pre_s, or pre_a and pre_b are available. All

transactions are required to be signed by both Alice and Bob.

Protocol Flow. We proceed exactly as in $\textsc{Contract}^{\mathrm{B}}$, except for the two additional transactions in $tx_{A^{\mathsf{A}}_{\mathsf{defuse}}}$ and $tx_{A^{\mathsf{A}}_{\mathsf{burn}}}$. Alice and Bob sign the two transactions prior to signing the setup transaction. Alice has the signatures on $tx_{A^{\mathsf{A}}_{\mathsf{defuse}}}$ that she keeps privately, while $tx_{A^{\mathsf{A}}_{\mathsf{burn}}}$ and the signatures on this transaction are broadcast to the network. After the setup transaction, $tx^{\mathsf{A}}_{\mathsf{stp}}$ is published on the blockchain, and the execution phase begins whenever $A^{\mathsf{A}}_{\mathsf{defuse}}$ is to be activated, Alice publishes $tx_{A^{\mathsf{A}}_{\mathsf{defuse}}}$ and the signatures on the blockchain. To activate $A^{\mathsf{A}}_{\mathsf{burn}}$, transaction $tx_{A^{\mathsf{A}}_{\mathsf{burn}}}$ along with the corresponding signatures, and either pre_a or pre_s are published on the blockchain. Notice that, as required, only one of these two transactions can be posted, allowing us to realize that $A^{\mathsf{A}}_{\mathsf{defuse}}$ and $A^{\mathsf{A}}_{\mathsf{burn}}$ are mutually exclusive. The rest of the protocol proceeds as the description for the previous instantiation.

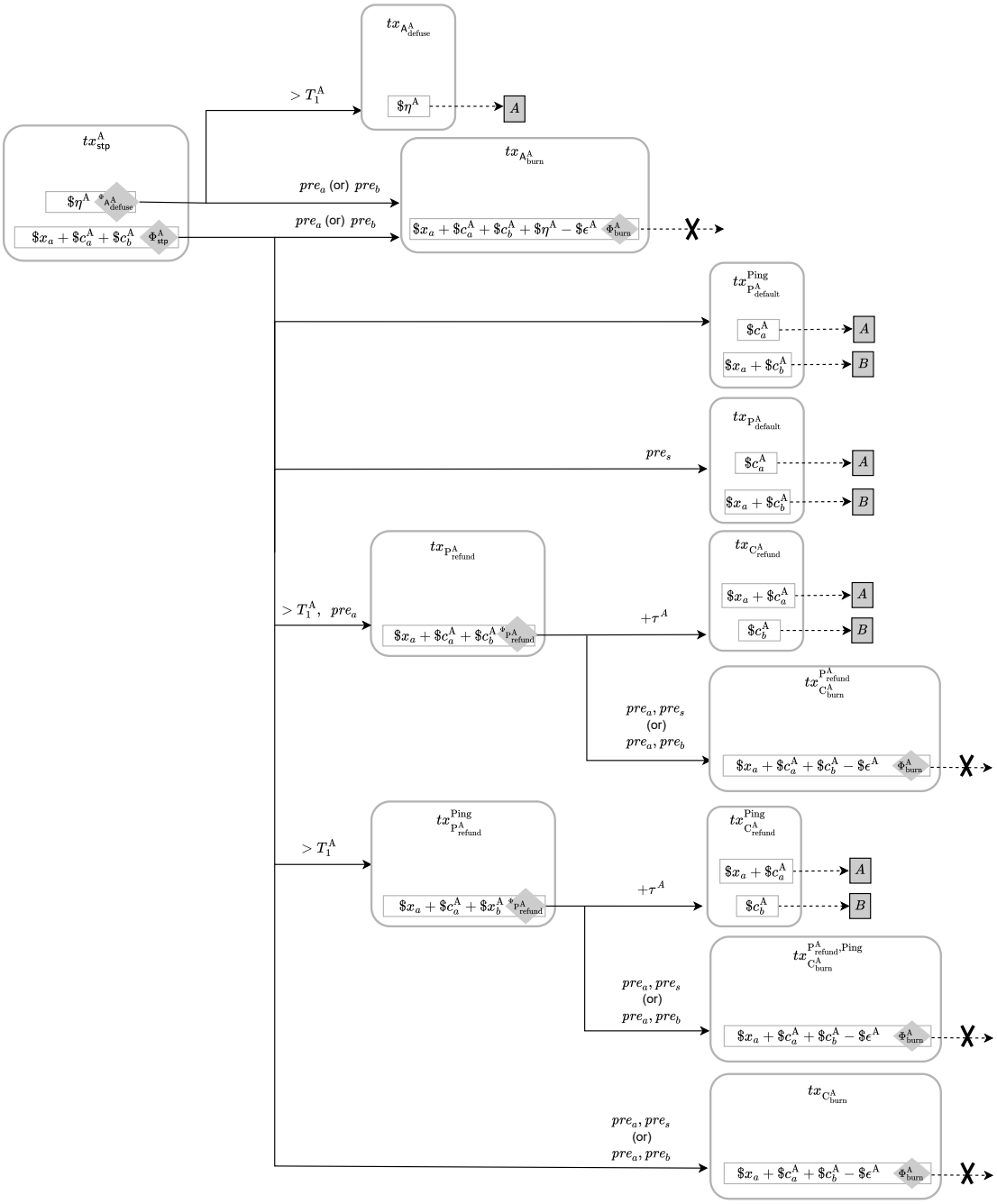

Figure 8: The transaction flow of CONTRACT$^{\mathrm{A}}$ in Bitcoin for an atomic swap with bounded maximin fairness. Rounded boxes denote transactions, and rectangles within are outputs of the transaction. Incoming arrows denote transaction inputs, outgoing arrows denote how an output can be spent by a transaction at the end of the arrow. Solid lines indicate the transaction output can be spent only if both users sign the spending transaction. Dashed arrows indicate that the output can be spent by one user (A for Alice, and B for Bob). The timelock (T_1^{A} and τ^{A}) associated with a transaction output is written over the corresponding outgoing arrow.

References

[Bon16] Joseph Bonneau. Why buy when you can rent? - bribery attacks on bitcoin-style consensus. In *FC Workshops*, 2016.

[CGJ+17] Arka Rai Choudhuri, Matthew Green, Abhishek Jain, Gabriel Kaptchuk, and Ian Miers. Fairness in an unfair world: Fair multiparty computation from public bulletin boards. In *ACM CCS*, 2017.

[CGL+18] Kai-Min Chung, Yue Guo, Wei-Kai Lin, Rafael Pass, and Elaine Shi. Game theoretic notions of fairness in multi-party coin toss. In *TCC*, volume 11239, pages 563–596, 2018.

[CMST22a] Hao Chung, Elisaweta Masserova, Elaine Shi, and Sri AravindaKrishnan Thyagarajan. Ponyta: Foundations of side-contract-resilient fair exchange. *Cryptology ePrint Archive*, 2022.

[CMST22b] Hao Chung, Elisaweta Masserova, Elaine Shi, and Sri AravindaKrishnan Thyagarajan. Rapidash: Foundations of side-contract-resilient fair exchange. Cryptology ePrint Archive, 2022.

[CRS24] Hao Chung, Tim Roughgarden, and Elaine Shi. Collusion-resilience in transaction fee mechanism design. In *Proceedings of the 25th ACM Conference on Economics and Computation*, pages 1045–1073, 2024.

[CS23] Hao Chung and Elaine Shi. Foundations of transaction fee mechanism design. In *SODA*, 2023.

[DH20] Apoorvaa Deshpande and Maurice Herlihy. Privacy-preserving cross-chain atomic swaps. In *International conference on financial cryptography and data security*, pages 540–549. Springer, 2020.

[Her18] Maurice Herlihy. Atomic cross-chain swaps. In *PODC*, 2018.

[MD19] Mahdi H. Miraz and David C. Donald. Atomic cross-chain swaps: Development, trajectory and potential of non-monetary digital token swap facilities. In *AETiC*, 2019.

[MHM18] Patrick McCorry, Alexander Hicks, and Sarah Meiklejohn. Smart contracts for bribing miners. In *FC Workshops*, 2018.

[MMS+] Giulio Malavolta, Pedro Moreno-Sanchez, Clara Schneidewind, Aniket Kate, and Matteo Maffei. Anonymous multi-hop locks for blockchain scalability and interoperability. In *NDSS 2019*.

[ora22] Oracle manipulation attacks are rising, creating a unique concern for defi. 2022.

[PG99] Henning Pagnia and Felix C. Gartner. On the impossibility of fair exchange without a trusted third party. Technical report, 1999.

[PS17] Rafael Pass and Elaine Shi. Fruitchains: A fair blockchain. In *PODC*, 2017.

[SCW23] Elaine Shi, Hao Chung, and Ke Wu. What can cryptography do for decentralized mechanism design? In *ITCS 2023*, 2023.

[TSW24] Sri Aravinda Krishnan Thyagarajan, Pratik Soni, and Ke Wu. Game-theoretically fair distributed sampling. In *CRYPTO (8)*, volume 14927 of *Lecture Notes in Computer Science*, pages 207–239. Springer, 2024.

[TYME21] Itay Tsabary, Matan Yechieli, Alex Manuskin, and Ittay Eyal. MAD-HTLC: because HTLC is crazy-cheap to attack. In *S&P*, 2021.

[vdM19] Ron van der Meyden. On the specification and verification of atomic swap smart contracts. In *IEEE ICBC*, 2019.

[WAS22] Ke Wu, Gilad Asharov, and Elaine Shi. A complete characterization of game-theoretically fair, multi-party coin toss. In *Eurocrypt*, 2022.

[WHF19] Fredrik Winzer, Benjamin Herd, and Sebastian Faust. Temporary censorship attacks in the presence of rational miners. In *IEEE European Symposium on Security and Privacy Workshops*, 2019.

[WSZN22] Sarisht Wadhwa, Jannis Stoeter, Fan Zhang, and Kartik Nayak. He-htlc: Revisiting incentives in htlc. Cryptology ePrint Archive, Paper 2022/546, 2022. https://eprint.iacr.org/2022/546.

[ZDBN19] Jean-Yves Zie, Jean-Christophe Deneuville, Jeremy Briffaut, and Benjamin Nguyen. Extending atomic cross-chain swaps. In *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, 2019.