Optimizing Liveness for Blockchain-Based Sealed-Bid Auctions in Rational Settings

Maozhou Huang[1], Xiangyu Su[2], Mario Larangeira[2,3], and Keisuke Tanaka[2]

[1] Department of Mathematics, New Uzbekistan University, 1 Movarounnahr street, Tashkent, Uzbekistan mz.huang@newuu.uz.
[2] Department of Mathematical and Computing Sciences, School of Computing, Institute of Science Tokyo. Tokyo-to Meguro-ku Oookayama 2-12-1 W8-55. su.x.4029@m.isct.ac.jp, tanaka.k.8268@m.isct.ac.jp.
[3] Input Output, Global. mario.larangeira@iohk.io.

Abstract. Blockchain-based auction markets offer stronger fairness and transparency compared to their centralized counterparts. Deposits and sealed bid formats are usually applied to enhance security and privacy. However, to our best knowledge, the formal treatment of deposit-enabled sealed-bid auctions remains lacking in the cryptographic literature. To address this gap, we first propose a decentralized anonymous deposited-bidding (DADB) scheme, providing formal syntax and security definitions. Unlike existing approaches that rely on smart contracts, our construction utilizes a mainchain-sidechain structure that is also compatible with the extended UTXO model. This design further allows us to develop a consensus mechanism on the sidechain dedicated to securely recording bids for allocation. Specifically, we build atop an Algorand-style protocol and integrate a novel block qualification mechanism into the block selection. Consequently, we prove, from a game-theoretical perspective, that our design optimizes liveness latency for rational users who want to join the auction, even without explicit incentives (e.g., fees) for including bids. Finally, our implementation results demonstrate the potential performance degradation without the block qualification mechanism.

Keywords: Deposit-Enabled Sealed-Bid Auctions · Blockchain-Based Auctions · Algorand-Style Consensus Protocol · Rational Analysis

1 Introduction

Auction markets have been a long-standing topic in applied economics. Decentralization enhances their fairness and transparency by eliminating the requirement for trusted auctioneers while ensuring security through the persistence and liveness of consensus protocols. Blockchain, as an embodiment of these protocols [2], is ideal for securely recording and verifying auction outcomes.

However, the direct application of blockchain to auction markets suffers from the maximal extractable value (MEV) activities. Specifically, miners can manipulate the inclusion of bids (similar to transactions in traditional cryptocurrency

blockchains) to maximize their profit (see [40, Section 2.2.2] for a precise definition). As demonstrated in [32], sealed-bid auctions address this issue by concealing both bid content and bidder information, preventing miners from learning bids' intent. Moreover, most existing implementations of blockchain-based sealed-bid auctions [11, 17, 19, 23, 30–32, 34, 39, 41, 44] rely on the Ethereum-type smart contracts [42], and some further assume a trusted execution environment [18,29]. These assumptions may increase execution costs in real life because fees for including such smart contracts are much higher than those for including normal transactions [26], even the extended UTXO-type ones [8].

In contrast, this work takes a more fundamental approach, building directly on the consensus layer with a committee-based Algorand-style blockchain. This enables us to incorporate a novel block qualification mechanism into block selection. We prove that our enhancements to the Algorand protocol [10] preserve its original security and, additionally, allow us to reduce liveness latency when rational users operate the protocol, even without fees for including bids.

1.1 Our Approach and Contributions

We propose a blockchain-based sealed-bid auction protocol (named Aucrand)[1], which consists of the following two parts.

A decentralized anonymous deposited-bidding scheme. For auction markets, a reasonable requirement for bidders is an appropriate amount of deposits before participation. This ensures bidders' compliance with the bidding process[2]. Moreover, for blockchain-based auctions, deposits are separated from bids to conceal bidders' information (hence to prevent MEV activities). This separation and, hence, a deposit-enabled sealed-bid auction can be constructed using an Ethereum-type smart contract that overlooks the state of deposits and bids [32].

Although we separate deposits from bids as in the protocol in loc. cit., we notice that, to the best of our knowledge, there is no cryptographic formalization of this separation. Following the spirit of the well-known Zerocoin [33] and decentralized anonymous credential [21] frameworks, we formalize the deposit-enabled sealed-bid auction into a decentralized anonymous deposited-bidding (DADB) scheme, providing its syntax and security (see Appendix B). As for the security: anonymity requires that no adversary can identify the owner of a given bid, even when provided with a set of deposits containing this bid's corresponding deposit; and one-more bidding unforgeability requires that no user can issue a bid without possessing an associated deposit transaction.

Assuming a secure ledger (*i.e.*, with persistence and liveness), we give a generic construction (see Section 3.1) for the DADB scheme based on a CCA-secure timed commitment scheme [25] and signatures of knowledge [9]. The timed

[1] Our protocol is also versatile enough to support various auction models, *e.g.*, double auctions and frequent batch auctions, as detailed in Appendix A.1.

[2] As noted in [12, Footnote 2], an auction with deposits that mismatch the value of bids may suffer from severe defaults.

feature is used to automatically open bids so that the results of bid allocation are publicly accessible. We prove, by reduction (in Lemma 1), that our construction satisfies anonymity and one-more bidding unforgeability. In our construction, we leverage the mainchain-sidechain structure, where the mainchain is solely for recording deposits. The deposit mechanism on the mainchain can be implemented using the extended UTXO model [8], offering more simplicity and lower execution cost than relying on the Ethereum-type smart contract [42].

A committee-based Algorand-style sidechain protocol. Given the structure above, we design the consensus mechanism for the sidechain to establish an agreement on the common set of (sealed) bids among all users. Our approach leverages an Algorand-style protocol. Concretely, based on one-more bidding unforgeability, we adapt Algorand's stake-based committee selection to our deposit setting and introduce a block qualification mechanism.

For the block qualification mechanism (see Eq. 3.1), in an arbitrary honest user's view, a candidate block is admitted in her block selection only when this candidate block contains $\geq \mathfrak{h} \cdot y$ bids, where $\mathfrak{h} \in (0, 1]$ is a prefixed threshold and y denotes the number of bids in the largest candidate block received. Then, among the admitted candidate blocks, this user determines the block based on the hash values corresponding to the candidate block. We note that if each honest user sets the largest candidate block as the block, then many candidate blocks proposed by honest users may have zero chance of being the block. For instance, the adversary may send its bids to some honest users only. Moreover, when an honest user counts the bids in a received candidate block, she only counts those bids appearing in both her mempool and the candidate block. This implies that the malicious users "have to propagate" their bids.

By assuming the honest majority of deposits, *i.e.*, $> 2/3$ of deposit transactions, are submitted to the mainchain by honest users, we prove in Theorem 1 (Section 4.1) that our modifications do not compromise the security of the original Algorand protocol [10]. On the other hand, our modification plays a critical role in the game theoretical analysis. In Section 4.2, we define a game that simulates the candidate block proposal phase of our sidechain protocol. Based on anonymity, this game replaces honest users with rational ones whose utility is derived solely from having their bids included in the block. We also provide an explicit attacker strategy in the game, which replicates the outcomes produced by the adversary, thereby validating that our game accurately simulates the block selection process. Our block qualification mechanism enables us to show that the honest behavior (*i.e.*, including all bids) constitutes an equilibrium, even without incentives such as fees for including bids (Theorem 2). Additionally, we present experimental results demonstrating the potential performance degradation caused by the selfish behavior of rational users (Section 5).

1.2 Related Works

A comparison with blockchain-based sealed-bid auction protocols is given below. Our literature review will focus on the committee-based consensus protocols

Table 1: Comparison with related works.

Works	Techniques	Focus of Analysis
[11, 23, 30, 31, 39, 41, 44]	SC & Secure blockchain	SC-based privacy and security
[18, 29]	SC & TEE	TEE-based security
[17, 19]	SC	SC programming
[34]	SC & State channel	SC-based dispute resolution
[32]	SC	Economical analysis
This work	EUTXO-based structure & Block qualification	DADB provable security; Consensus when honest/rational

that influenced our design. Protocols, *e.g.*, Algorand [10], Ouroboros [14, 27], and [13, 36, 37, 43] employ a small, randomly selected committee to make decisions, enhancing scalability and efficiency compared to traditional consensus models, *e.g.*, [28]. Ouroboros, a Nakamoto-style blockchain [2] with longer finality times, introduced a probabilistic, stake-based committee selection mechanism using verifiable random functions. This mechanism was later refined in the Byzantine agreement-based Algorand protocol [10]. By adapting Algorand's construction to our deposit setting, we inherit the resilience against adaptive corruption from its frequent committee selection and the rapid finality from the Byzantine agreement.

However, due to the lack of rational analysis, the impact of rational user behavior is uncertain in these protocols. Our approach addresses this gap by modifying Algorand's block proposal and selection. Through game-theoretic analysis, we prove that our modifications can effectively mitigate the negative impacts of rational behavior, maintaining protocol performance and security.

1.3 Organization

Section 2 defines the general protocol execution model and sketches the building blocks. The rest sections present our main contributions: Section 3 is devoted to our Aucrand protocol, consisting of a DADB construction and an Algorand-style sidechain protocol; We then analyze, in Section 4, the security of our protocol against the Byzantine adversary, and from a game theoretic perspective; Section 5 presents the implementation results.

2 Preliminaries

Notations. This paper uses κ for the security parameter. For any integer $a \leq b$, let $[a \,..\, b] \coloneqq \{a, a+1, \ldots, b\}$; and any integer $n > 0$, let $[n] \coloneqq [1 \,..\, n]$ and $[n]_0 \coloneqq [0 \,..\, n]$. $a \leftarrow \mathsf{Alg}$ denotes that a is assigned the output of the algorithm Alg on fresh randomness. Denote a collision-free hash function by $H\colon \{0,1\}^* \to \{0,1\}^\kappa$.

2.1 Protocol Execution Model

We adopt the standard Interactive Turing Machines (ITM) model [7], in which a protocol refers to algorithms for a set of nodes (users) to interact with each other. Regarding the adversary model, for security analysis in Section 4.1, we consider that all corrupted users are controlled by a *rushing* Byzantine adversary $\mathcal{A}$ who can read inputs and set outputs for these users. For game-theoretic analysis in Section 4.2, all users are modeled to be rational in a pre-defined game. Their behaviors are described by strategy sets and utility functions. Our additional protocol settings are as follows.

- Time and network: Round-based execution that is further divided into steps; All users' clocks proceed at the same speed, and the local computation is instant; A semi-synchronous network with two known delay upper bounds: λ for short messages; Λ for full blocks.
- Participation and corruption: A constrained permissionless setting: Permissionless so that anybody can submit deposit transactions; and Permissioned for the sidechain protocol, *i.e.*, only users who have deposited can issue bids; The adversary can corrupt honest users adaptively at any time.

2.2 Building Blocks

We treat the cryptographic primitives below as black-boxes. Detailed notations and formal definitions can be found in Appendix A.2.

- A digital signature scheme $\mathsf{DS} := (\mathsf{KGen}, \mathsf{Sign}, \mathsf{SigVrfy})$ satisfying correctness and EUF-CMA [22].
- A non-interactive timed commitment (NITC) scheme TC, in which commitments can be forced opened after time t_{fo}. Here,

$$\mathsf{TC} := (\mathsf{PGen}, \mathsf{Com}, \mathsf{OpenVrfy}, \mathsf{FOpen})$$

  satisfies correctness, CCA-hiding and CCA-binding [25].
- A zero-knowledge proofs of knowledge (ZKPoK) protocol that satisfies completeness, (perfect) zero-knowledge, and knowledge-soundness [15].

Briefly, we denote the key pair of user i by (sk_i, pk_i). Additionally, we consider the ephemeral keys model from [10], which supports forward security [3, 24], to achieve resilience against adaptive corruption. The ephemeral key pair in round r step s is denoted by $(sk_i^{r,s}, pk_i^{r,s})$. Let $sig_i(m)$ ($esig_i(m)$) be the (ephemeral) signature on message m. We write $SIG_i(m) := (m, sig_i(m))$ and $ESIG_i(m) := (m, esig_i(m))$ for the message-signature pair in the rest of this paper.

For ZKPoK, we consider the protocols tailored to proofs of set membership and range proofs [5]. We refer to the non-interactive proofs, obtained by Fiat-Shamir heuristic [16], as signatures of knowledge as given in [9]. We adopt the notations from [6]. For the commit algorithm Com of any commitment scheme and a value x, let $\mathsf{NIZKPoK}\{(c, r) : (c, \cdot) \leftarrow \mathsf{Com}(x; r) \wedge c \in C\}$ denote a set membership proof that proves the knowledge of witness r *s.t.* $(c, \cdot) \leftarrow \mathsf{Com}(x; r) \wedge c \in C$. We denote the signature of knowledge on message m *w.r.t.* this relation by

$\mathsf{SoK}[m]\{(c,r):(c,\cdot)\leftarrow\mathsf{Com}(x;r)\wedge c\in C\}$. Moreover, for a range proof concerning $[a,b]$ where $a,b\in\mathbb{R}$, we denote the proof of x *s.t.* $(c,\cdot)\leftarrow\mathsf{Com}(x)\wedge x\in[a,b]$ by $\mathsf{NIZKPoK}\{(x):(c,\cdot)\leftarrow\mathsf{Com}(x)\wedge x\in[a,b]\}$.

A secure public ledger. To simplify the design of our mainchain-sidechain structure, we assume a secure public ledger protocol $(\Pi_{\mathsf{M}},\mathcal{L})$ to be our "mainchain" for recording deposit transactions. We consider it to satisfy 0-persistence and u-liveness. This can be achieved by truncating the last k blocks from any k-persistent and (k,u)-live ledger protocol (see Appendix A.2 for properties' formal definitions). For notations, we omit Π_{M} and use $\mathcal{L}$ to refer to both the protocol and the ledger. Let $\mathcal{L}[-1]$ denote the head (*i.e.*, the latest block) of $\mathcal{L}$.

3 Our Aucrand Protocol

To avoid defaults, we require users to submit deposit transactions to the mainchain to participate in the auction market. Each deposit transaction recorded on the mainchain allows its owner to issue exactly one bid in the sidechain protocol. All bids should be sealed for better privacy guarantees and to mitigate the MEV activity. Our modified Algorand-style sidechain protocol is then executed to achieve consensus on the sealed bids for all honest users, which is represented by the resulting blockchain. We employ a timed-release cryptographic primitive, allowing users to seal bids and reveal them in a timely manner. Then, the allocation results will be automatically determined based on the revealed bids and the predefined auction model.

This section presents our main contribution: the Aucrand protocol. It includes the construction of our proposed decentralized anonymous deposited-bidding (DADB) scheme and a detailed explanation of the associated sidechain.

3.1 A Generic DADB Construction

Our DADB scheme (see Appendix B.1 for syntax) formalizes the deposit-enable sealed-bid auction. We present its construction based on the mainchain-sidechain structure. On the mainchain $\mathcal{L}$, each user i with a key pair (sk_i,pk_i) from DS is uniquely identified by her public key pk_i. A participant holding multiple key pairs is regarded as multiple users. However, the total amount of deposit transactions a participant can submit is upper bounded by her currency on the ledger $\mathcal{L}$.

For simplicity, we regard the bidding string in DADB syntax to be a normalized bidding price: $P\in[-1,0)\cup(0,1]$, where the sign of P indicates the direction of bids, *i.e.*, $P>0$ for buy bids, $P<0$ for sell bids, and the actual bidding price is $|P|$. For each bid, a user is required to make a constant amount of deposit $d>1$ on $\mathcal{L}$ by submitting a transaction that embeds a commitment obtained from a unique serial number. A time interval, counted by the number of blocks on $\mathcal{L}$, for submitting deposit transactions is called a deposit epoch, indexed by $e\geq 0$. Hereby, our construction is performed for each epoch and starts with $\mathsf{Setup}(1^\kappa)\rightarrow(\mathcal{L},e,d)$.

A Deposit Epoch.

Let TC be an NITC scheme. In deposit epoch $e \geq 0$ on $\mathcal{L}$, a user i with (sk_i, pk_i) performs $(\mathsf{Deposit}, \mathsf{dVrfy}, \mathsf{ReadL})$.

- $\mathsf{Deposit}(e, d, sk_i)$ samples a serial number and a randomness with $S, r \xleftarrow{\$} \{0,1\}^\kappa$. It outputs a deposit transaction $\mathsf{d}_i = SIG_i(e, d, c)$ to $\mathcal{L}$, where $(c, \cdot) \leftarrow \mathsf{TC.Com}(S; r)$ is the commitment *w.r.t.* (S, r).
- $\mathsf{dVrfy}(e, d, pk_j, \mathsf{d}_j)$ parses $\mathsf{d}_j = SIG_j(e', d', c)$. It outputs 1 if $e' = e \wedge d' = d \wedge \mathsf{SigVrfy}(pk_j, \mathsf{d}_j) = 1$; or 0 otherwise.
- $\mathsf{ReadL}(\mathcal{L}, e)$ is only executed after the liveness[a] delay u of $\mathcal{L}$ following the end of the deposit epoch e. Let PK^e denote the set of all public keys on $\mathcal{L}$ in epoch e. ReadL outputs:
  - For each $j \in PK^e$, the set of valid deposit transactions issued by j: $D_j^e := \{\mathsf{d}_j \mid \mathsf{dVrfy}(e, d, pk_j, \mathsf{d}_j) = 1\}$;
  - The set
$$C^e := \left\{ c \in \mathsf{d} \,\middle|\, \mathsf{d} \in \bigcup_{j \in PK^e} D_j^e \right\}$$
  of all committed serial numbers[b].

[a] For completeness, we provide the interface for reading the ledger in addition to our DADB syntax. Moreover, the liveness of $\mathcal{L}$ guarantees that any d outputted to $\mathcal{L}$ will be recorded after this delay.

[b] Due to the one-to-one correspondence between a deposit transaction and a committed serial number, we will focus on using C^e in the following of this paper for better explicitness.

Note that d serves only as a data structure for recording deposits. It must be completed according to the accounting model of $\mathcal{L}$, *e.g.*, the (extended) UTXO [1, 8] or the account model [42]. We deliberately consider *signed* deposit transactions to align with the transaction format of UXTO models, contrasting with the unsigned deposit payloads proposed in the smart contract-based [32].

In parallel to a deposit epoch, a user can issue bids *w.r.t.* deposit transactions submitted by herself. Each bid is sealed in the sense that: (1) the price P is committed with an NITC scheme while guaranteed by an NIZKPoK range proof π_P showing $|P| \in (0, 1]$; (2) the bid is decoupled from its corresponding deposit transaction (also the user's public key) using an SoK scheme. Even if a user only knows a subset $C' \subseteq C^e$ when issuing bids, she can still hide her deposit transactions by proving, in a zero-knowledge manner, that she knows the randomness *s.t.* the committed serial number lies in C'. The bidding process is specified as follows.

Bidding Process.

Let TC be an NITC scheme. To issue and verify bids, a user performs $(\mathsf{Bidding}, \mathsf{bVrfy})$ as follows.

- $\mathsf{Bidding}(P, S, r, C')$ takes as input a bidding price P, a serial number S, a randomness r, and a set of committed serial numbers $C' \subseteq C^e$ chosen by the user. It outputs a sealed bid $\mathsf{b} := (c_P, \pi_P, S, \pi_{\mathsf{SoK}})$ where:
  - $(c_P, \cdot) \leftarrow \mathsf{TC.Com}(P)$ is the commitment of P and can be forced opened via $\mathsf{TC.FOpen}$.
  - $\pi_P = \mathsf{NIZKPoK}\{(P) : (c_P, \cdot) \leftarrow \mathsf{TC.Com}(P) \wedge |P| \in (0, 1]\}$ is a range proof showing $|P| \in (0, 1]$ *w.r.t.* c_P.
  - $\pi_{\mathsf{SoK}} = \mathsf{SoK}[(c_P, \pi_P)]\{(c, r) : (c, \cdot) \leftarrow \mathsf{TC.Com}(S; r) \wedge c \in C'\}$ is a signature of knowledge on (c_P, π_P) where $C' \subseteq C^e$.
- $\mathsf{bVrfy}(\mathsf{b}, C')$ takes as input a sealed bid $\mathsf{b} = (c_P, \pi_P, S, \pi_{\mathsf{SoK}})$ and a set of committed serial numbers C'. It outputs 1 if π_P is a valid proof of c_P indicating $|P| \in (0, 1]$, and π_{SoK} is a valid signature of knowledge on (c_P, π_P) indicating that the commitment of S is in C'; or 0 otherwise. Specifically, we have $\mathsf{bVrfy}(\mathsf{b}, C^e) = 1$ if $\exists C' \subseteq C^e$ *s.t.* $\mathsf{bVrfy}(\mathsf{b}, C') = 1$.

Moreover, we adapt the rigorous security model from [33] to our bidding setting, and define three properties in Appendix B.2. Briefly:

- Correctness means that any output from $\mathsf{Deposit}$ passes dVrfy, and any bid from $\mathsf{Bidding}$ corresponding to a valid deposit transaction passes bVrfy.
- Anonymity requires that no adversary can distinguish any pair of bids not issued by her, given the deposit transactions corresponding to the bids.
- One-more bidding-unforgeability indicates that no adversary can produce $m+1$ valid bids with no duplicated serial number when given a set of deposit transactions and some of their corresponding bids, and allowed to submit m deposit transactions.

We conclude with the following Lemma. See Appendix B.3 for a detailed proof.

Lemma 1. *Assuming a secure signature* DS *and a secure public ledger* $\mathcal{L}$ *(with u-liveness delay), our construction satisfies the following properties.*

- *Correctness (Definition* 9*).*
- *Anonymity (Definition* 10*) if the NITC scheme* TC *is CCA-hiding and the signature of knowledge proof* π_{SoK} *is at least computationally zero-knowledge.*
- *One-more bidding-unforgeability (Definition* 11*) if the NITC scheme* TC *is CCA-binding, the signature of knowledge proof* π_{SoK} *is at least computationally zero-knowledge and is knowledge-sound.*

3.2 The Sidechain Protocol

Based on the deposit transactions recorded on the mainchain $\mathcal{L}$, users seek to agree on a common set of sealed bids to determine bid allocation results. Unlike existing smart contract-based solutions, we design a dedicated sidechain for

the consensus of sealed bids. The sidechain is based on an Algorand-style (*i.e.*, Byzantine agreement) protocol [10] with modified block proposal and selection.

For simplicity, the following considers a fixed epoch e and omits it in upper indices, *i.e.*, we use PK, D_i, and C to denote respectively PK^e, D_i^e, and C^e.

Initialization. When a deposit epoch is completely finished, *i.e.*, $(\{D_i\}_{i\in PK}, C) \leftarrow \mathsf{ReadL}(\mathcal{L}, e)$ is known to all users after the liveness delay of $\mathcal{L}$, users initialize the sidechain protocol with an empty blockchain $\mathcal{C} = \emptyset$ at round index $r = 0$ and step index $s = 1$. Let pp be the parameter consisting of the tuple $(\lambda, \Lambda, \mathcal{L}, e, d, \{D_i\}_{i\in PK}, C)$ and[3]:

- $\mathfrak{h} \in [0, 1)$, the threshold for qualifying candidate blocks. See Eq. 3.1.
- L, analyzed in Section 4.1, is the lower bound for the number of serial numbers in candidate blocks so that the block qualification mechanism is activated.
- $p^{r,s}$ is the fraction whose denominator is $|C|$, and the numerator is the expected number of votes for committee members[4] in round $r \geq 0$ step $s \geq 1$.
- t_H denotes the number of votes needed to certify a block.

We parameterize the protocol with pp and denote it by $(\Pi^{\mathsf{pp}}, \mathcal{C} = \emptyset)$. Thereafter, pp is in the input of all algorithms and will be omitted for simplicity.

Moreover, when specified with a round index $r \geq 0$, $\mathcal{C}^r \coloneqq B^0||\cdots||B^r$ denotes the blockchain by the end of round r, where $B^{r'}$ is the selected block of round r' for all $r' \in [r]_0$. The seed of round r is used to determine the committee selection for the next round and is computed by $Q^r = H(SIG_{\ell^r}(Q^{r-1}), r)$. We allow $r = -1$, and put $\mathcal{C}^{-1} = B^{-1} \coloneqq \mathcal{L}[-1]$ and $Q^{-1} \coloneqq H(\mathcal{L}[-1])$. The user who generates round r's selected block is called the leader of round r, denoted by ℓ^r. Thereafter, by leader selection, we mean the block selection (we use the former to align with "committee selection" and "potential leader").

Definition 2. A *candidate block* generated by user i in round $r \geq 0$ of the sidechain protocol $(\Pi^{\mathsf{pp}}, \mathcal{C}^{r-1})$ is defined as

$$B_i^r \coloneqq (r, H(B^{r-1}), SIG_i(Q^{r-1}), \mathsf{B}_i^r),$$

where B_i^r is a set of bids collected by i. If the leader selection fails in round r, $B^r = B_\epsilon^r \coloneqq (r, H(B^{r-1}), Q^{r-1}, \emptyset)$; otherwise, $B^r = B_{\ell^r}^r$ for the leader ℓ^r.

Committee selection. Our committee selection adapts the stake-based mechanism [10, Section 6] to the set of deposits $\bigcup_{i\in PK} D_i$. Given pp, a user i is selected with a weight proportional to $a_i \coloneqq |D_i|$. See Appendix C.1 for details. Briefly:

- $\sigma_i^{r,s} \coloneqq SIG_i(r, s, Q^{r-1})$, i's round r step s credential for committee selection.
- $\mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s})$ outputs the number of i's votes (Algorithm 1).
- $\mathsf{GetMinHash}(a_i, \sigma_i^{r,1})$ essentially outputs the "minimum hash" (Algorithm 2).

[3] We can configure Λ for each round r *s.t.* it is lower bounded by the number of serial numbers in C but not in the bids on $\mathcal{C}^{r-2}$ ($\mathcal{C}^{r-2}$ is known to all users).

[4] A deposit transaction becomes one vote in committee selection with probability $p^{r,s}$.

– Let $SV^{r,s}$ denote the set of committees in round $r \geq 0$ step $s \geq 1$. Moreover, (1) $i \in SV^{r,s}$ if and only if $\mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s}) > 0$. Particularly, we call users in $SV^{r,1}$ *potential leaders*, and those in $SV^{r,s}$ for $s > 1$ *verifiers*. (2) i's power as a committee member is proportional to the number of her votes.

Step-by-step execution. This section presents the step 1 and 2 in the sidechain protocol Π^{pp}, in which we make significant modifications to the original Algorand. For completeness, the remaining steps are provided in Appendix C.3.

In any round $r \geq 0$, each user i starts her (own) round r as soon as she is sure about B^{r-1}, *i.e.*, gets $CERT^{r-1}$ consisting of Q^{r-1} and $\geq t_H$ signatures on the same $H(B^{r-1})$ from the same step (see step 5)[5].

Step 1: Block Proposal.

Each user i starts her step 1 as soon as she starts round r. She waits[a] time $t_1 \coloneqq \Lambda$ and performs as follows then.

1. i gets her mempool MP_i^r by collecting bids b from the network[b] *s.t.*:
   (a) $\mathsf{bVrfy}(\mathsf{b}, C) = 1$;
   (b) $\nexists\, \mathsf{b}' \in \mathsf{MP}_i^r \cup \big(\bigcup_{r' \in [r-1]_0} \mathsf{B}^{r'}\big)$ satisfying $\mathsf{b}' \neq \mathsf{b}$ and the serial number S in both b and b'.
2. i runs $\mathsf{GetVotes}(p^{r,1}, a_i, \sigma_i^{r,1}) = x$. She ends her step 1 if $x = 0$ (*i.e.*, $i \notin SV^{r,1}$); Otherwise, she performs as follows[c].
   i sets $\mathsf{B}_i^r = \mathsf{MP}_i^r$, computes her candidate block $B_i^r = (r, H(B^{r-1}), SIG_i(Q^{r-1}), \mathsf{B}_i^r)$, prepares $m_i^{r,1} \coloneqq (B_i^r, esig_i(H(B_i^r)), \sigma_i^{r,1})$ with her ephemeral key pair $(sk_i^{r,1}, pk_i^{r,1})$, destroys $sk_i^{r,1}$, and propagates $m_i^{r,1}$.

[a] The waiting period ensures that the user can receive B^{r-1}. Moreover, we propose an alternative configuration in Appendix D that eliminates this waiting procedure, provided we admit that a serial number appears at most twice.
[b] Some of these bids may come from candidate blocks in previous rounds.
[c] Unlike Algorand [10], each user in $SV^{r,1}$ propagates the entire candidate block, as our leader selection requires the qualification of candidate blocks.

We prepare $\mathsf{VrfyMsg}$ (Algorithm 3) for users to verify the validity of messages. Specially for step 1, the algorithm also verifies candidate blocks, *e.g.*, each bid in the input candidate block should be valid and not duplicated concerning its serial number. Let $M_i^{r,s}$ denote the set of all *valid* messages collected by user i from $SV^{r,s}$. That is, $\mathsf{VrfyMsg}(p^{r,s}, a_j, \mathcal{C}^{r-1}, m_j^{r,s}) = 1$ for any $m_j^{r,s} \in M_i^{r,s}$.

For each user i, the leader selection takes as input her mempool MP_i^r and the valid message set $M_i^{r,1}$ collected by her. We outline $\mathsf{SelectL}(\mathsf{MP}_i^r, M_i^{r,1})$ algorithm for this purpose, which is specified in Algorithm 4.

On a high level, the algorithm first extracts all serial numbers from MP_i^r to a set of serial numbers, denoted by SP_i^r. By one-more bidding-unforgeability, we

[5] Some user may not know the full block B^{r-1} even she knows $CERT^{r-1}$ which is a short message.

consider SP_i^r instead of MP_i^r. Hence, the adversary cannot take advantage by issuing multiple bids under the same serial number. Next, from the candidate block B_j^r embedded in each message $m_j^{r,1} \in M_i^{r,1}$, it extracts all serial numbers to a set of serial numbers, denoted by S_j^r. It sets $V_i \coloneqq \{\mathsf{S}_j^r \mid m_j^{r,1} \in M_i^{r,1}\}$,

$$y \coloneqq \max_{\mathsf{S}_j^r \in V_i} \{|\mathsf{S}_j^r \cap \mathsf{SP}_i^r|\}, \quad W_i \coloneqq \begin{cases} \{j \mid |\mathsf{S}_j^r \cap \mathsf{SP}_i^r| \geq \hbar y, \mathsf{S}_j^r \in V_i\} & y \geq L; \\ \{j \mid \mathsf{S}_j^r \in V_i\} & y < L. \end{cases} \tag{3.1}$$

The intersection defining y will be discussed in Remark 15. $\mathsf{SelectL}$ outputs the leader in the view of i as $\ell \leftarrow \arg\min_{j \in W_i} \mathsf{GetMinHash}(a_j, \sigma_j^{r,1})$.

Step 2: Leader Selection.

In any round $r \geq 0$, each user i starts her step 2 as soon as she finishes her step 1. The user waits time $\lambda + \Lambda$ in step 2. Hence, the total waiting time is $t_2 \coloneqq \lambda + 2\Lambda$. She performs as follows *after* the waiting period.

1. i collects $M_i^{r,1}$. If $M_i^{r,1} = \emptyset$, she sets $v_i \coloneqq \perp$; Otherwise, she selects $\ell \coloneqq \mathsf{SelectL}(\mathsf{MP}_i^r, M_i^{r,1})$ and sets $v_i \coloneqq (H(B_\ell^r), \ell)$.
2. i runs $\mathsf{GetVotes}(p^{r,2}, a_i, \sigma_i^{r,2}) = x$. She stops and propagates nothing if $x = 0$ (*i.e.*, $i \notin SV^{r,2}$); Otherwise, she prepares[a] a message $m_i^{r,2} \coloneqq (ESIG_i(v_i), \sigma_i^{r,2}, SIG_\ell(Q^{r-1}))$ with her ephemeral key pair $(sk_i^{r,2}, pk_i^{r,2})$, destroys $sk_i^{r,2}$, and propagates $m_i^{r,2}$.

[a] $SIG_\ell(Q^{r-1})$ is included in $m_i^{r,2}$ to propagate the seed in her view.

4 Security Analysis for Aucrand

In this section, we first show that Aucrand is secure under certain assumptions (cf. those in [10, Section 5.2]). Then, we define a strategic game simulating Aucrand step 1 and work out its equilibrium. Retain the notations in Section 3.2.

4.1 Security Against the Byzantine Adversary

We first model a user to be either honest or corrupted. The corrupted ones (named malicious users) are controlled by a Byzantine adversary. We make the honest majority of deposit (HMD) assumption, *i.e.*, the fraction of the deposit transactions submitted by honest users (in each round of the sidechain protocol) is *always* $> 2/3$ (although there might be corruption). By the one-more bidding-unforgeability property (Lemma 1), HMD is equivalent to the honest majority of serial number (HMS) assumption, *i.e.*, the fraction of serial numbers corresponding to deposit transactions submitted by the honest users is always $> 2/3$. We list notations and assumptions below (cf. those in [10, Section 5]):

- h is a number *s.t.* the fraction of the deposit transactions submitted by honest users is always $> h$.

– $HSV^{r,s}$ and $MSV^{r,s}$ denotes respectively the subset of $SV^{r,s}$ of honest verifiers and malicious verifiers. $|HSV^{r,s}|$ and $|MSV^{r,s}|$ denote respectively the number of votes from verifiers in $HSV^{r,s}$ and $MSV^{r,s}$: for $x_i \coloneqq \mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s})$,

$$|HSV^{r,s}| \coloneqq \sum_{i \in HSV^{r,s}} x_i \text{ and } |MSV^{r,s}| \coloneqq \sum_{i \in MSV^{r,s}} x_i.$$

– The numbers $p^{r,s}$ and t_H are chosen so that we have the following inequality with overwhelming probability

$$|HSV^{r,s}|/2 + |MSV^{r,s}| < t_H < |HSV^{r,s}|. \tag{4.1}$$

– T^{r+1} denotes the time when the first honest user is sure about B^r *i.e.*, got $CERT^r$. I^r denotes the interval $[T^r, T^r + \lambda]$.
– ℓ denote the *r-leader* who, if exists[6], is the unique user *s.t.* the value $(H(B^r_\ell), \ell)$ gets $\geq t_H$ votes from $HSV^{r,2}$. This uniqueness follows from that no other value gets $\geq t_H$ votes from $SV^{r,2}$ (by Lemma 3(1) below).

Lemma 3 (Proved in Appendix E.1)**.** (1) *(cf.* [10, Lemma 5.7]*) Assume that*[7] *all honest users are sure about the same B^{r-1}. Let v denote the value in the message signed by verifiers in $SV^{r,s}$. If there is a value v getting $\geq t_H$ votes from verifiers in $SV^{r,s}$, then there exists* no *other value $v' \neq v$ s.t. v' and v have the same length and v' gets $\geq t_H$ votes from verifiers in $SV^{r,s}$.*
(2) *Assume that all honest users are sure about the same B^{r-1} in the time interval I^r and the same B^{r-2}. Any honest user i receive the block B^{r-1} before collecting her mempool MP_i^r of round r.*

Qualification in the leader selection. Admit that all honest users are sure about the same B^{r-1}. For a verifier $i \in HSV^{r,2}$, given the set W_i in Eq. 3.1, the leader in her view is the user in W_i who has the minimal hash. For an honest potential leader $i' \in HSV^{r,1}$, we know $\mathsf{S}_{i'}^r = \mathsf{SP}_{i'}^r$. However, HMS does not necessarily imply that $\mathsf{SP}_{i'}^r$ is large enough[8] so that it may happen that $i' \notin W_i$. In the leader selection of round r, one of the following cases happens:
– (bad case) $\exists\, i \in HSV^{r,2}$ and $\exists\, i' \in HSV^{r,1}$ *s.t.* $i' \notin W_i$.
– (good case) otherwise.
We propose basic properties concerning two cases (see Appendix E.1 for proofs).

Proposition 4. (1) *If the good case happens and some honest potential leader ℓ has the minimal hash, then ℓ is the r-leader.*
(2) *Put $\mathsf{SP}_H^r \coloneqq \bigcup_{i \in HSV^{r,2}} \mathsf{SP}_i^r$. If $\geq \mathfrak{h}$ of serial numbers in SP_H^r belong to $\mathsf{SP}_{i'}^r$ for any verifier $i' \in HSV^{r,1} \cup HSV^{r,2}$, then the good case happens.*

[6] Even if no r-leader exists, the block may be nonempty and the leader for round r may exist, *i.e.*, $B^r \neq B^r_\epsilon$.
[7] This assumption is the induction hypothesis for proving Theorem 1.
[8] In fact, if malicious users do not send their bids to i' before honest users collect their mempools, then $\mathsf{SP}_{i'}^r$ may be too small so that the candidate block of i' gets disqualified by honest users who have received malicious users' bids.

(3) *Admit* $\mathfrak{h} = 2/3$. *Assume that*[9] *all honest users are sure about the same* $B^{r'-1}$ *in* $I^{r'}$ *for* $r' \in [r, r+5]$. *If* $L \geq |C| \cdot 5\%$, *where* C *denotes the set of all serial numbers, then the bad case can not consecutively happen*[10] *for* $r' \in [r..r+5]$.

Remark 5. (1) Due to HMS, the good case could be easily realized if all honest users issue their bids before the round 0 of the side chain.

(2) By the inverse of Proposition 4(2), to let the bad case happen, some malicious users have to send a large enough number of their bids to a part of honest users so that $< \mathfrak{h}$ of serial numbers in SP_H^r belong to $\mathsf{SP}_{i'}^r$ for some verifier $i' \in HSV^{r,1} \cup HSV^{r,2}$. We use this to show Proposition 4(3).

(3) The occurrence of the bad case does not necessarily imply that the leader is malicious. We may assume that the adversary intends to let some malicious user's candidate block get $\geq t_H$ votes from $SV^{r,2}$, because for otherwise, no honest user in step $s \geq 3$ regards a malicious user as the leader. Then, by the intersection in Eq. 3.1, the malicious users need to compete with the hash values of $> 1/2$ of honest potential leaders (see Remark 15).

The main theorem. We propose a sketch of the main theorem for the security of Aucrand with HMD, whose details are postponed to Appendix E.2. This theorem is similar to that of [10, Theorem 1] in Algorand. In fact, HMD plays a similar role as the honest majority of money assumption in [10].

Theorem 1 (A sketch)**.** *The following properties hold with overwhelming probability for each round* $r \geq 0$:

(1) *All honest users agree on the same block* B^r. *Each bid* b *in* B^r *satisfies* $\mathsf{bVrfy}(\mathsf{b}, C) = 1$ *and that any* b' *in the chain* $\mathcal{C}^r$ *with* $\mathsf{b}' \neq \mathsf{b}$ *corresponds to a serial number different from that of* b. *Moreover, for the number* z *of bids issued by honest users and not in the chain*[11]*, if* B^r *is nonempty and* $z \geq L$, *then* B^r *contains at least* $\mathfrak{h}z$ *bids.*

(2) *If r-leader* ℓ *exists, then all honest users are sure about* B^r *generated by* ℓ *in the time interval* I^{r+1} *and* $T^{r+1} \leq T^r + 5\lambda + 2\Lambda$.

(3) *If r-leader does not exist, then all honest users are sure about* B^r *in the time interval* I^{r+1} *and* $T^{r+1} \leq T^r + (6L^r + 8)\lambda + 2\Lambda$. *Here* L^r *denotes the random variable representing the number of Bernoulli's trials needed to see a* 1.

(4) *If the good case happens in rounds* $r-1$ *and* r, *then the probability of some honest user becoming r-leader is* $\geq h^2(1 + h - h^2)$.

4.2 Security in Rational Settings

When the context is clear, we put $H^s \coloneqq HSV^{r,s}$ and $M^s \coloneqq MSV^{r,s}$ for $s \geq 2$.

[9] By Theorem 1, this assumption is fulfilled in our setting.

[10] Admit a slightly lower $\mathfrak{h}$, *e.g.*, $\mathfrak{h} = 1/2$. If $L \geq |C| \cdot 1.1\%$, then the bad case can not consecutively happen for $r' \in [r..r+5]$.

[11] In fact, one may replace the number z of bids issued by honest users with the number of serial numbers belonging to the serial number pool SP_i^r for any honest user i.

An extensive game. The goal is to define a game to simulate Aucrand step 1. On a high level, we replace honest potential leaders with those rational users who want to join the auction.

Recall that we assume a network with bounded delay and the adversary being rushing, *i.e.*, can see all candidate blocks of potential leaders beforehand. Hence, it is reasonable to treat the Aucrand step 1 as an extensive game with perfect information. For simplicity, we only present a strategic game below. One may regard it as the strategic form of an extensive game given in Appendix E.3.

Definition 6 (r-Election as a strategy game)**.** The strategic game r-election is the tuple $\left(SV^{r,1}, (S_i)_{i\in SV^{r,1}}, (u_i)_{i\in SV^{r,1}}\right)$ defined as follow:

(1) *The set of players* $SV^{r,1}$ consists of potential leaders of Aucrand round r. A player is said to be an *r-candidate* if she is an honest user before waiting time Λ ends in round r. Let I denote the set of *r-candidates*. Put $J \coloneqq SV^{r,1} \setminus I$. A player $j \in J$ is called an *r-villain*.

(2) For each r-candidate $i \in I$, her *strategy set* S_i is the set $\mathcal{MP}_i^r$ of all subsets of MP_i^r. Let $j \in J$ be an r-villain. Let A_j denote the set consisting of functions

$$f_j : PK \setminus \{j\} \to \mathcal{MP}_j^r.$$

We may regard that $\exists\, i, i' \in PK$ with $i \neq i'$ such that $f_j(i) \neq f_j(i')$. The j's *strategy set* S_j consists of functions $s_j : \prod_{i\in I} S_i \to A_j$.

(3) Regard $s_i \in S_i$ as the candidate block propagated by the r-candidate i. Given $(s_i)_{i\in I}$, for $j \in J$ and $s_j \in S_j$, regard $f_j(i')$ with $f_j = s_j((s_i)_{i\in I})$ as the candidate block that j sends to the user $i' \in PK \setminus \{j\}$. Following Eq. 3.1, each verifier i'' in H^2 can select a leader according to the received candidate blocks.
   - If an r-candidate $\hat{i}$ is the leader whose candidate block gets $\geq t_H$ votes from verifiers in H^2 (we name $\hat{i}$ the *r-leader*[12]), we put $u_i = n_i$ and $u_j = -|s_{\hat{i}}|$ for $i \in I$ and $j \in J$, where n_i denotes the number of bids issued by i that are included in the candidate block of $\hat{i}$.
   - Otherwise, put $u_i = -\tau$ for $i \in I$ and $u_j = 0$ for $j \in J$, where $\tau > 0$ reflects[13] the time cost caused by empty blocks.

The rationale of u_i: By anonymity (Lemma 1), bids' information is hidden. For an r-candidate, once there are some bids in the pool, her allocation result depends on the price of her bids and whether or not these bids are included in the block. Moreover, the r-candidate's time cost for obtaining this result increases if there are more empty blocks. So, within one epoch, in her view:

(1) whether a bid *not* issued by her is included in the chain does not affect her allocation result;

(2) she earns nothing if her bids are not included;

[12] The r-leader here is compatible with the one defined in Section 4.1. We restrict to the case where an r-leader is an r-candidate (cf. utility function of an r-villain).

[13] $\tau > 0$ is a relative value for the time cost, which is the difference between the time cost of empty blocks and that of the case where an r-leader exists.

(3) the relative time cost of empty blocks should give her a negative utility.
Therefore, she is better off letting her own bids in the chain and avoiding the empty block case.

The rationale of u_j:

(1) By the utility function, r-villains' goal is to prevent that some r-candidate becomes the r-leader (cf. Corollary 21(1));
(2) As r-villains have the same utility, we may regard that they are controlled by one player, denoted *r-Villain*. In particular, her strategy set is

$$(s_j)_{j\in J} : \prod_{i\in I} S_i \to \prod_{j\in J} A_j \tag{4.2}$$

and r-Villain's utility is u_j for any $j \in J$.

Remark 7. By the equality $u_i = -\tau$, even if all of an r-candidate's bids are included in the chain, she is better off avoiding the empty block case.

Remark 8 (r-Villain behaves like adversary). Admit: (A.1) The r-villains are controlled by the adversary in round r step $s \geq 2$; (A.2) If not corrupted, r-candidates behave the same as honest users after round r step 1. We show (see Corollary 21) that the strategies of the r-villains may lead to the same consequences as those caused by the adversary (cf. Theorem 1(3)) if Aucrand step 1 replaced with the r-election. This means that we make virtually no assumption about r-villains' behavior. Hence, we regard r-villains to behave the same as malicious users and the r-Villain to resemble the Byzantine adversary.

On equilibrium of the r-election game. We work out an equilibrium under certain assumptions. For a player $k \in SV^{r,1}$, put $-k \coloneqq SV^{r,1} \setminus \{k\}$. Retain the notations in Definition 6. We admit, *w.l.o.g.*[14], $\mathsf{SP}_j^r = \mathsf{SP}_H^r \coloneqq \bigcup_{i\in H^2} \mathsf{SP}_i^r$ for any $j \in J$. For $i \in I$ and $j \in J$, the inclusion $\mathsf{SP}_i^r \subseteq \mathsf{SP}_j^r$ holds as honest users propagate their bids.

Consider a strategy profile $(s_k^*) = (s_k^*)_{k\in SV^{r,1}} \in \prod_{k\in SV^{r,1}} S_k$ defined below.

– Each r-candidate $i \in I$ behaves the same as an honest user, *i.e.*, $s_i^* = \mathsf{MP}_i^r$.
– For an r-villain $j \in J$, consider a constant function s_j^* on $\prod_{i\in I} S_i$ *s.t.*

$$s_j^* : PK \setminus \{j\} \to \mathcal{MP}_j^r,\ s_j^*(i) = \begin{cases} \mathsf{MP}_j^r & i \in H_+^2; \\ \emptyset & i \in H_-^2, \end{cases}$$

where $H_+^2 \subsetneq H^2$ consists of players $i \in H^2$ *s.t.* $|H_+^2| = a$ for $|H^2| - t_H < a < t_H$ and $H_-^2 \coloneqq H^2 \setminus H_+^2$ hold. Here $|H_+^2|$ is calculated as in Section 4.1.

Theorem 2. *Assume that $\geq \mathfrak{h}$ of series numbers in SP_H^r belong to SP_i^r for any $i \in I \cup H^2$ and $\mathfrak{h} \cdot |\mathsf{SP}_H^r| \geq L$. Then the profile (s_k^*) is an equilibrium for the r-election. Moreover, this profile is an equilibrium even if we regard all r-villains as one player – the r-Villain.*

[14] This assumption is natural. Indeed, it implies $\mathsf{SP}_j^r = \mathsf{SP}_{j'}^r$ for any $j, j' \in J$, which is compatible with that the r-Villain controls r-villains (Eq. 4.2). Moreover, by Eq. 3.1, for an r-villain j, the case $\mathsf{SP}_j^r \supset \mathsf{SP}_H^r$ and the case $\mathsf{SP}_j^r = \mathsf{SP}_H^r$ are equivalent.

The assumption of this theorem has appeared in Proposition 4(2). By Proposition 4(3) (see also Remark 5(2)), the adversary can not take advantage of the negation of this assumption consecutively, *i.e.*, it can not let the bad case happen consecutively. In this theorem, when regarding all r-villains as the r-Villain, the utility function is referred to the one defined in Eq. 4.2. The proof of this theorem is postponed to Appendix E.4.

Remark 9. We sketch partial results concerning this theorem's enhancements below (see Lemmas 22, 23, and Proposition 24 for details):

(1) For an r-candidate i, the strategy MP_i^r is a best response to any strategy of $-i$ if there exists another r-candidate chooses a strategy containing MP_i^r.[15]
(2) For $i \in I$, any strategy $s_i \subsetneq \mathsf{MP}_i^r$ is not a best response to certain $-i$'s strategy.
(3) Regard all r-villains as one player. Given certain behavior of the adversary before the r-election, the strategy profile (s_k^*) is not a subgame equilibrium.

Let h be a number such that $\geq h$ fraction of deposit transactions are issued by honest users (hence candidates) in any round. Theorem 1(4) implies

Corollary 10. *Assume that the setting and the equilibrium in the theorem hold for rounds $r' \in [r-1..r+k]$, $k \geq 1$. Then if a serial number S belongs to SP_i^r for all r-candidates i, then the probability for a bid corresponding to S included in the chain $\mathcal{C}^{r+k}$ is at least $1-(1-p_h)^k$ with $p_h = h^2(1+h-h^2)$.*

Remark 11 (Fee-less liveness). The liveness of blockchain protocols [20] briefly means that transactions known by all honest users are eventually inserted into the chain before a certain amount of delay. The corollary may read: without explicit reward, *e.g.*, fees for including bids, the bids known by all rational users who want to join the auction are eventually inserted into the sidechain before a certain amount of delay under certain conditions.

5 Sidechain Implementation

The implementation[16] is for the sidechain protocol *without* our block qualification mechanism Eq. 3.1. Instead, we use the minimum hash-based block selection. Each network node is either:

(1) an honest user who includes all bids received into their candidate block;
(2) a selfish rational user as in Remark 8(A.2), *i.e.*, only cares if her own bids are included in the chain (cf. Definition 6(3)).

Our experiment aims to exhibit the potential harm caused by selfish behavior without the mechanism. We exclude malicious users from our experiment as they can only cause a limited number of empty blocks (see Theorem 1 and Remark 8).

[15] We tried to remove the condition "$\geq \mathfrak{h}$ of serial numbers in SP_H^r belong to SP_i^r for any $i \in I \cup H^2$" to get a stronger result. This is difficult, but we give a partial result.
[16] See `https://anonymous.4open.science/r/aucrand` for the code.

5.1 Setup and Metrics

The experiment considers a single deposit epoch. We first set up 10 users based on Python's thread-based parallelism. Then, consider the case where they produce 5000 deposit transactions (hence 5000 serial numbers) within this deposit epoch. The sidechain protocol is set to run for 100 rounds. Each user issues 5 bids per round (hence 50 bids from 10 users) and receives bids from all other users in the network. The rational user fraction, denoted by R, varies from 0 to 100%, *i.e.*, from 0 to 10 rational users. For the set of bids B^r in block B^r of round $r \in [99]_0$, we define metric functions on r:

- The inclusion rate: $IR = \left|\mathsf{B}^r\right|/(\#\text{bids issued in round } r) = \left|\mathsf{B}^r\right|/50$;
- The cumulative inclusion rate: $CIR = \sum_{r' \in [r]_0} \left|\mathsf{B}^{r'}\right|/(50 \cdot (r+1))$;
- The mempool size: $MS = 50 \cdot (r+1) - \sum_{r' \in [r]_0} \left|\mathsf{B}^{r'}\right|$;
- The accumulated weights: $\sum IR^2$, the sum extends over blocks with $IR > 1$.[17]

5.2 Results and Analysis

Figure 1(a-c) (*resp.*, (d)) illustrate the variation of IR, CIR, MS (*resp.*, $\sum IR^2$) throughout protocol execution and under different R (*resp.*, across R). The figures demonstrate the apparent inefficiencies introduced by selfish behavior in block generation: If $R = 0$, *i.e.*, all users are honest, the selected block should include all issued bids in each round, resulting in $IR = 1$ and $CIR = 1$ (the blue line). If $R > 0$, then due to the selfish behavior, blocks proposed by honest users may include bids from previous rounds, leading to $IR > 1$; and bids may not be included in time, leading to $CIR < 1$ and $MS > 0$. When $R < 100\%$, as R increases, the accumulated weight of big blocks $\sum IR^2$ increases (because $\sum IR^2$ is dominated by IR, which indicates the size of blocks). Overall, the inefficiency becomes more severe as R increases.

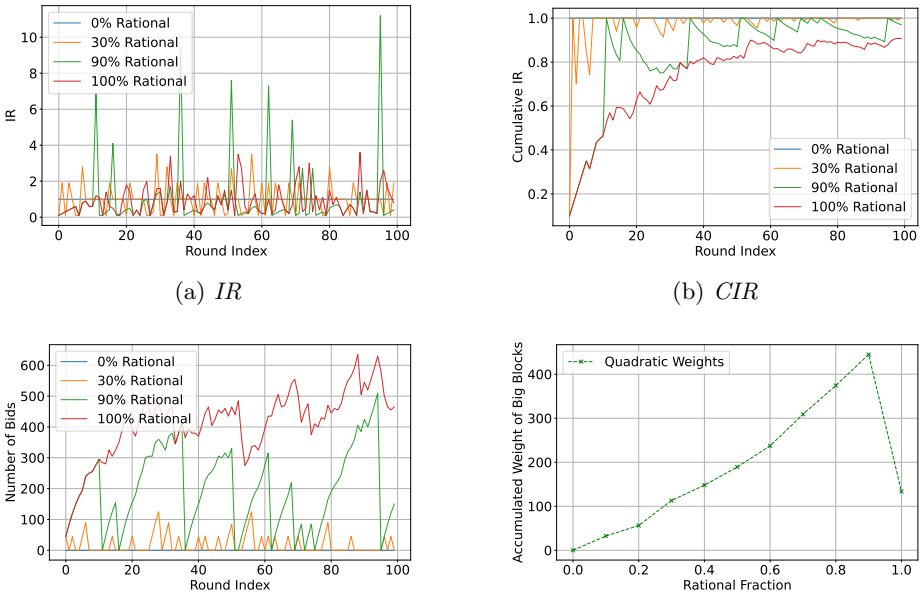
In the case of $R = 100\%$, IR and $\sum IR^2$ drop compared to the $R = 90\%$ case due to the absence of honest users. In the red line of Figure 1a, $IR > 1$ indicates some bids of a user have been waiting for at least 10 rounds.

Consequently, without our block qualification mechanism, these inefficiencies would result in the performance degradation of the blockchain:

(1) leaving some bids unprocessed and not included by the blockchain in time;
(2) potentially causing delays and congestion as larger blocks take longer to propagate through the network.

Implications for other protocols. Our experiments reconfirm the importance of incentivizing honest behavior among rational users. Conventional solutions rely on explicit rewards, such as transaction fees, to encourage honest behavior. These rewards may impose additional costs on participants (*e.g.*, for trading). In contrast, by incorporating the novel block qualification mechanism, Aucrand leverages the natural incentives provided by users' own interests in sealed-bid

[17] Such blocks are regarded as big blocks, taking longer to propagate in the network.



(a) *IR* (b) *CIR*

(c) *MS*. (d) Accumulated big blocks weights.

Fig. 1: Results across rational user fractions.

auctions to incentivize honest behavior, hence requiring no extra fees on its users.

6 Conclusion

In this work, we propose Aucrand, a novel blockchain-based sealed-bid auction protocol. Our contributions include the formalization of a decentralized anonymous deposited-bidding (DADB) scheme and the development of an Algorand-style sidechain. By integrating a novel block qualification mechanism into the consensus process, we first prove that our protocol maintains Algorand's original security guarantees. Regarding rational users, our implementation results show that their behavior can lead to performance degradation. However, through game-theoretical analysis, we demonstrate that our block qualification mechanism can incentivize honest behavior, effectively mitigating such degradation, even *without* relying on explicit incentives.

References

1. Atzei, N., Bartoletti, M., Lande, S., Zunino, R.: A formal model of bitcoin transactions. In: Meiklejohn, S., Sako, K. (eds.) FC 2018. LNCS, vol. 10957, pp. 541–560. Springer, Heidelberg (Feb / Mar 2018). https://doi.org/10.1007/978-3-662-58387-6_29
2. Barbulescu, R., Gaudry, P., Kleinjung, T.: The tower number field sieve. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015, Part II. LNCS, vol. 9453, pp. 31–55. Springer, Heidelberg (Nov / Dec 2015). https://doi.org/10.1007/978-3-662-48800-3_2
3. Bellare, M., Miner, S.K.: A forward-secure digital signature scheme. In: Wiener, M.J. (ed.) CRYPTO'99. LNCS, vol. 1666, pp. 431–448. Springer, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-48405-1_28
4. Budish, E., Cramton, P., Shim, J.: The High-Frequency Trading Arms Race: Frequent Batch Auctions as a Market Design Response *. The Quarterly Journal of Economics **130**(4), 1547–1621 (07 2015). https://doi.org/10.1093/qje/qjv027, https://doi.org/10.1093/qje/qjv027
5. Camenisch, J., Chaabouni, R., shelat, a.: Efficient protocols for set membership and range proofs. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 234–252. Springer, Heidelberg (Dec 2008). https://doi.org/10.1007/978-3-540-89255-7_15
6. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups (extended abstract). In: Kaliski Jr., B.S. (ed.) CRYPTO'97. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (Aug 1997). https://doi.org/10.1007/BFb0052252
7. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd FOCS. pp. 136–145. IEEE Computer Society Press (Oct 2001). https://doi.org/10.1109/SFCS.2001.959888
8. Chakravarty, M.M.T., Chapman, J., MacKenzie, K., Melkonian, O., Jones, M.P., Wadler, P.: The extended UTXO model. In: Bernhard, M., Bracciali, A., Camp, L.J., Matsuo, S., Maurushat, A., Rønne, P.B., Sala, M. (eds.) FC 2020 Workshops. LNCS, vol. 12063, pp. 525–539. Springer, Heidelberg (Feb 2020). https://doi.org/10.1007/978-3-030-54455-3_37
9. Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 78–96. Springer, Heidelberg (Aug 2006). https://doi.org/10.1007/11818175_5
10. Chen, J., Micali, S.: Algorand: A secure and efficient distributed ledger. Theor. Comput. Sci. **777**, 155–183 (2019). https://doi.org/10.1016/J.TCS.2019.02.001
11. Constantinides, T., Cartlidge, J.: Block auction: A general blockchain protocol for privacy-preserving and verifiable periodic double auctions. In: Xiang, Y., Wang, Z., Wang, H., Niemi, V. (eds.) 2021 IEEE Blockchain 2021, Melbourne, Australia, December 6-8, 2021. pp. 513–520. IEEE (2021). https://doi.org/10.1109/BLOCKCHAIN53845.2021.00078
12. Cramton, P.C.: Money out of thin air: The nationwide narrowband pcs auction. Journal of Economics & Management Strategy **4**(2), 267–343 (1995). https://doi.org/https://doi.org/10.1111/j.1430-9134.1995.00267.x, https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1430-9134.1995.00267.x
13. Daian, P., Pass, R., Shi, E.: Snow white: Robustly reconfigurable consensus and applications to provably secure proof of stake. In: Goldberg, I., Moore, T. (eds.) FC 2019. LNCS, vol. 11598, pp. 23–41. Springer, Heidelberg (Feb 2019). https://doi.org/10.1007/978-3-030-32101-7_2

14. David, B., Gazi, P., Kiayias, A., Russell, A.: Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 66–98. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78375-8_3
15. Feige, U., Shamir, A.: Zero knowledge proofs of knowledge in two rounds. In: Brassard, G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 526–544. Springer, Heidelberg (Aug 1990). https://doi.org/10.1007/0-387-34805-0_46
16. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987). https://doi.org/10.1007/3-540-47721-7_12
17. Galal, H.S., Youssef, A.M.: Verifiable sealed-bid auction on the ethereum blockchain. In: Zohar, A., Eyal, I., Teague, V., Clark, J., Bracciali, A., Pintore, F., Sala, M. (eds.) FC Workshop. LNCS, vol. 10958, pp. 265–278. Springer (2018). https://doi.org/10.1007/978-3-662-58820-8_18
18. Galal, H.S., Youssef, A.M.: Trustee: Full privacy preserving vickrey auction on top of ethereum. In: Bracciali, A., Clark, J., Pintore, F., Rønne, P.B., Sala, M. (eds.) FC Workshop. LNCS, vol. 11599, pp. 190–207. Springer (2019). https://doi.org/10.1007/978-3-030-43725-1_14
19. Galal, H.S., Youssef, A.M.: Publicly verifiable and secrecy preserving periodic auctions. In: Bernhard, M., Bracciali, A., Gudgeon, L., Haines, T., Klages-Mundt, A., Matsuo, S., Perez, D., Sala, M., Werner, S. (eds.) FC Workshop. LNCS, vol. 12676, pp. 348–363. Springer (2021). https://doi.org/10.1007/978-3-662-63958-0_29
20. Garay, J.A., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: Analysis and applications. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 281–310. Springer, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46803-6_10
21. Garman, C., Green, M., Miers, I.: Decentralized anonymous credentials. In: NDSS 2014. The Internet Society (Feb 2014)
22. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing **17**(2), 281–308 (Apr 1988)
23. Górski, T., Bednarski, J.: Modeling of smart contracts in blockchain solution for renewable energy grid. In: Moreno-Díaz, R., Pichler, F., Quesada-Arencibia, A. (eds.) 17th Computer Aided Systems Theory - EUROCAST 2019, Las Palmas de Gran Canaria, Spain, February 17-22, 2019, Part I. LNCS, vol. 12013, pp. 507–514. Springer (2019). https://doi.org/10.1007/978-3-030-45093-9_61
24. Itkis, G., Reyzin, L.: Forward-secure signatures with optimal signing and verifying. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 332–354. Springer, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_20
25. Katz, J., Loss, J., Xu, J.: On the security of time-lock puzzles and timed commitments. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part III. LNCS, vol. 12552, pp. 390–413. Springer, Heidelberg (Nov 2020). https://doi.org/10.1007/978-3-030-64381-2_14
26. Khyathi, Dalal: Ethereum gas fees are sky high: Here are some social media reactions. `https://www.nasdaq.com/articles/ethereum-gas-fees-are-sky-high:-here-are-some-social-media-reactions` (2023), Nasdaq
27. Kiayias, A., Russell, A., David, B., Oliynykov, R.: Ouroboros: A provably secure proof-of-stake blockchain protocol. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 357–388. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63688-7_12

28. Lamport, L., Shostak, R., Pease, M.: The byzantine generals problem. ACM Transactions on Programming Languages and Systems pp. 382–401 (July 1982), https://www.microsoft.com/en-us/research/publication/byzantine-generals-problem/
29. Liu, B., Yang, Y., Wang, R., Hong, Y.: Poster: Privacy preserving divisible double auction with A hybridized tee-blockchain system. In: 41st IEEE ICDCS 2021, Washington DC, USA, July 7-10, 2021. pp. 1144–1145. IEEE (2021). https://doi.org/10.1109/ICDCS51616.2021.00128
30. Liu, L., Du, M., Ma, X.: Blockchain-based fair and secure electronic double auction protocol. IEEE Intell. Syst. **35**(3), 31–40 (2020). https://doi.org/10.1109/MIS.2020.2977896
31. Ma, X., Xu, D., Wolter, K.: Blockchain-enabled feedback-based combinatorial double auction for cloud markets. Future Gener. Comput. Syst. **127**, 225–239 (2022). https://doi.org/10.1016/J.FUTURE.2021.09.009
32. McMenamin, C., Daza, V., Fitzi, M., O'Donoghue, P.: Fairtradex: A decentralised exchange preventing value extraction. In: Proceedings of the 2022 ACM CCS Workshop on Decentralized Finance and Security. p. 39–46. DeFi'22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3560832.3563439, https://doi.org/10.1145/3560832.3563439
33. Miers, I., Garman, C., Green, M., Rubin, A.D.: Zerocoin: Anonymous distributed E-cash from Bitcoin. In: 2013 IEEE Symposium on Security and Privacy. pp. 397–411. IEEE Computer Society Press (May 2013). https://doi.org/10.1109/SP.2013.34
34. Nguyen, T.D.T., Thai, M.T.: A blockchain-based iterative double auction protocol using multiparty state channels. CoRR (2020), https://arxiv.org/abs/2007.08595
35. Osborne, M., Rubinstein, A.: A Course in Game Theory. A Course in Game Theory, MIT Press (1994), https://mitpress.mit.edu/9780262650403/a-course-in-game-theory/
36. Pass, R., Shi, E.: The sleepy model of consensus. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part II. LNCS, vol. 10625, pp. 380–409. Springer, Heidelberg (Dec 2017). https://doi.org/10.1007/978-3-319-70697-9_14
37. Pass, R., Shi, E.: Thunderella: Blockchains with optimistic instant confirmation. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 3–33. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78375-8_1
38. Shi, Z., de Laat, C., Grosso, P., Zhao, Z.: Integration of blockchain and auction models: A survey, some applications, and challenges. Commun. Surveys Tuts. **25**(1), 497–537 (jan 2023). https://doi.org/10.1109/COMST.2022.3222403, https://doi.org/10.1109/COMST.2022.3222403
39. Thakur, S., Breslin, J.G., Malik, S.: Privacy-preserving energy trade using double auction in blockchain offline channels. In: Prieto, J., Martínez, F.L.B., Ferretti, S., Guardeño, D.A., Nevado-Batalla, P.T. (eds.) 4th IEEE BLOCKCHAIN 2022, L'Aquila, Italy, 13-15 July 2022. Lecture Notes in Networks and Systems, vol. 595, pp. 289–302. Springer (2022). https://doi.org/10.1007/978-3-031-21229-1_27
40. Weintraub, B., Torres, C.F., Nita-Rotaru, C., State, R.: A flash(bot) in the pan: measuring maximal extractable value in private pools. In: Proceedings of the 22nd ACM Internet Measurement Conference. p. 458–471. IMC '22, Association for Computing Machinery, New York, NY, USA (2022).

https://doi.org/10.1145/3517745.3561448, https://doi.org/10.1145/3517745.3561448
41. Wongsamerchue, T., Leelasantitham, A.: An electronic double auction of prepaid electricity trading using blockchain technology. J. Mobile Multimedia **18**(6), 1829–1850 (2022). https://doi.org/10.13052/JMM1550-4646.18616
42. Wood, G., et al.: Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper **151**(2014), 1–32 (2014)
43. Yin, M., Malkhi, D., Reiter, M.K., Golan-Gueta, G., Abraham, I.: HotStuff: BFT consensus with linearity and responsiveness. In: Robinson, P., Ellen, F. (eds.) 38th ACM PODC. pp. 347–356. ACM (Jul / Aug 2019). https://doi.org/10.1145/3293611.3331591
44. Zhang, S., Miao, P., Wang, B., Dong, B.: A privacy protection scheme of microgrid direct electricity transaction based on consortium blockchain and continuous double auction. IEEE Access **7**, 151746–151753 (2019). https://doi.org/10.1109/ACCESS.2019.2946794

A Auxiliary Definitions

This section introduces definitions for auctions and cryptographic primitives.

A.1 Two Sealed-Bid Auction Models

In this section, rather than "users," we use sellers and buyers in DA or traders in FBA. By "a bid", we mean a bid or an ask made by a seller, a buyer, or a trader.

Auction model design (market design) is a significant topic in economics. A successful auction design, e.g., the FCC spectrum auction (See [12]), allocates the goods to the bidder who values the goods most and maximizes the sellers' expected revenue.

Our motivation for this paper is to develop a protocol for realizing sealed-bid blockchain auctions. For the sake of explicit and instrumental, we are to concentrate on two prominent auction models: double auction (DA) and frequent batch auction (FBA). This subsection is devoted to a brief review of two models. In Section 3, we will propose a protocol, which realizes these two auction models.

Double auction. DA has been widely applied in blockchain-auctions [38, Section E]. In the DA, buyers pay money for one type of goods, and sellers provide goods in exchange for money. The DA consists of the following steps and is executed iteratively:

(1) This step is executed during a chosen period.
   - Buyers make bids, and sellers make asks. Each bid is a price-quantity pair (p, q) which means "buy q goods with the price of each good being p". Each ask is similarly defined.
   - The buyers and sellers submit their bids and asks to a trusted third party (known as an auctioneer). All bids and asks are sealed, *i.e.*, the submission is made privately, and the information of a bid or an ask is only known to the buyer or seller who submitted it.

(2) At the end of the chosen period, the trusted third party clears the market by drawing the demand and supply functions, and setting the uniform market-clearing price to the intersection point of the two functions. Then, the auctioneer rations bids or asks at this price to enable market clearing.

Frequent batch auction. FBA is proposed in [4] to resolve the high-frequency trading arms race[18]. In FBA, a trader may pay money for shares of stock and may sell her shares of stock for money. FBA briefly consists of the following steps and is executed iteratively:

[18] High-frequency trading (HFT) is a type of algorithmic trading. One feature of HFT is its very short-term investment horizon. As HFT is a continuous-time trading method, the execution time of HFT orders can decrease arbitrarily. The trading arms race between traders occurs as the trader who can execute her algorithms and orders faster takes more advantage.

(1) This step is executed within a round. Here, a round is a certain interval, named batch interval in [4]. As FBA models high-frequency trading, the length of batch intervals may be required to be short.
   – A trader may make bids as a buyer or make asks as a seller. A bid or ask might be a limit order or market order. A limit order is a price-quantity pair. A market order is a quantity. It is executed with the price so that this order can be executed immediately as a limit order.
   – The traders submit their orders to the auctioneer. The submission is made privately, and the information on an order is only known to the trader who submitted it.

(2) At the end of this round, the auctioneer clears the market in the same way as in Step (2) of DA. The orders that are not executed will remain and will be available in the next round unless it is canceled by the trader who submitted them.

A.2 Formal Definitions for Cryptographic Primitives

Digital signature schemes. A digital signature scheme is a tuple of algorithms $\mathsf{DS} := (\mathsf{KGen}, \mathsf{Sign}, \mathsf{SigVrfy})$ *s.t.*:

– $\mathsf{KGen}(1^\kappa)$ takes as input the security parameter κ and outputs a secret and public key pair (sk, pk). A key pair specified to user i is denoted by (sk_i, pk_i). We write the remaining algorithms specified to i.
– $\mathsf{Sign}(sk_i, m)$ outputs a signature $sig_i(m)$ on the message m under sk_i.
– $\mathsf{SigVrfy}(pk_i, m, sig_i(m))$ outputs 1 if $sig_i(m)$ is a valid signature on m concerning pk_i; or 0 otherwise.

We require DS to satisfy correctness and EUF-CMA [22].

Definition 1 (Correctness). *A signature scheme is correct if the following property holds for any $\kappa > 0$ and $(sk, pk) \leftarrow \mathsf{KGen}(1^\kappa)$.*

$$\Pr\left[\mathsf{SigVrfy}(pk, m, \mathsf{Sign}(sk, m)) = 1\right] = 1.$$

Definition 2 (EUF-CMA). *A signature scheme is EUF-CMA if for any adversary $\mathcal{A}$ that can query a signing oracle $\mathcal{O}_{\mathsf{Sign}}(sk, \cdot)$ for at most $q \leq \mathsf{poly}(\kappa)$ times s.t. the queried messages form a set Q, the following probability is $\mathsf{negl}(\kappa)$ for any $\lambda > 0$ and $(sk, pk) \leftarrow \mathsf{KGen}(1^\kappa)$.*

$$\Pr\left[m^* \notin \mathsf{Q} \wedge \mathsf{SigVrfy}(pk, m^*, sig(m)^*) = 1 \;\middle|\; (m^*, sig(m)^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(pk)\right]$$

Similar notations and security definitions are applied to the ephemeral keys model. As in Algorand, this model will be queried for all steps in a round. Specifically, a user i generates her ephemeral key pair with $(sk_i^{r,s}, pk_i^{r,s}) \leftarrow \mathsf{KGen}(1^\kappa, r, s)$ for round r and step s. She produces a signature $esig_i(m) \leftarrow \mathsf{Sign}(sk_i^{r,s}, m)$ *s.t.* $\mathsf{SigVrfy}(pk_i^{r,s}, m, esig_i(m)) = 1$. The concept of forward security [3, 24] extends the standard EUF-CMA security by ensuring that no PPT adversary can forge

a valid message-signature pair for any previous ephemeral key pair, even if the latest ephemeral key pair is compromised[19].

Timed commitment schemes. Following the formalization in [25], a t_{fo}-NITC scheme consists of a tuple of algorithms $\mathsf{TC} := (\mathsf{PGen}, \mathsf{Com}, \mathsf{OpenVrfy}, \mathsf{FOpen})$:

- $\mathsf{PGen}(1^\kappa)$ takes as input the security parameter κ and outputs a common reference string crs.
- $\mathsf{Com}(\mathsf{crs}, m)$ takes as input a string crs and a message m. It outputs a commitment c and an open o.
- $\mathsf{OpenVrfy}(\mathsf{crs}, c, m, o)$ takes as input a string crs , a commitment c, a message m, and an open o. It outputs 1 (accept) or 0 (reject).
- $\mathsf{FOpen}(\mathsf{crs}, c)$ takes as input a string crs and a commitment c. After time *at least* t_{fo}, it outputs m or $\perp$.

The security of NITC is enhanced to the CCA-model, in which the adversary is given access to a forced-opening oracle $\mathcal{O}_{\mathsf{FOpen}}(\mathsf{crs}, \cdot)$. Formally, the correctness, CCA-hiding, and CCA-binding are defined as follows.

Definition 3 (Correctness). *An NITC is correct if the following property holds for any $\kappa > 0$, $\mathsf{crs} \leftarrow \mathsf{PGen}(1^\kappa)$, and any m.*

$$\Pr\left[\begin{array}{l}\mathsf{OpenVrfy}(c, m, o) = 1 \wedge \\ \mathsf{FOpen}(c) = m\end{array} \middle| \; (c, o) \leftarrow \mathsf{Com}(m)\right] = 1$$

Definition 4 (CCA-Hiding). *An NITC is (t_1, t_2)-CCA-hiding if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ s.t. $\mathcal{A}_1$ can query a forced-opening oracle $\mathcal{O}_{\mathsf{FOpen}}(\mathsf{crs}, \cdot)$ for at most $t_1(\kappa)$ times, and $\mathcal{A}_2$ can query $\mathcal{O}_{\mathsf{FOpen}}(\mathsf{crs}, \cdot)$ for at most $t_2(\kappa)$ times without querying the challenged c, the following probability is $\mathsf{negl}(\kappa)$ for any $\kappa > 0$ and $\mathsf{crs} \leftarrow \mathsf{PGen}(1^\kappa)$.*

$$\left|\Pr\left[b' = b \middle| \begin{array}{l}(m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{FOpen}}}(\mathsf{crs}); \\ b \xleftarrow{\$} \{0,1\}; \\ (c, \cdot) \leftarrow \mathsf{Com}(m_b); \\ b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{FOpen}}}(c, \mathsf{st})\end{array}\right] - \frac{1}{2}\right|$$

Definition 5 (CCA-binding). *An NITC is t-CCA-binding if for any PPT adversary $\mathcal{A}$ that can query a forced-opening oracle $\mathcal{O}_{\mathsf{FOpen}}(\mathsf{crs}, \cdot)$ for at most t times, the following probability is $\mathsf{negl}(\kappa)$ for any $\kappa > 0$ and $\mathsf{crs} \leftarrow \mathsf{PGen}(1^\kappa)$.*

$$\Pr\left[\begin{array}{l}(c, m, o, m', o') \\ \quad \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{FOpen}}}(\mathsf{crs})\end{array} \middle| \begin{array}{l}\big(m \neq m' \wedge \mathsf{OpenVrfy}(c, m, o) = 1 \wedge \\ \quad \mathsf{OpenVrfy}(c, m', o') = 1\big) \vee \\ \big(\mathsf{OpenVrfy}(c, m, o) = 1 \wedge \mathsf{FOpen}(c) \neq m\big)\end{array}\right]$$

[19] The ephemeral keys model is also referred to as the erasure model in cryptography literature. Namely, an honest user can securely erase her ephemeral keys immediately after signing for a step. This ensures that: after the user signs a message with ephemeral keys (sk, pk), even if the user is corrupted instantly, the adversary cannot sign under sk.

NIZKPoK and SoK. Recall a general ZKPoK protocol. Let $\mathcal{L}_{\mathcal{R}} = \{x \mid \exists w \ s.t. \ (x,w) \in \mathcal{R}\} \subseteq \{0,1\}^*$ be a formal language *w.r.t.* a binary, polynomial-time (witness) relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$. Let $|x|$ denote the length of x. Then when given a polynomial-size (of $|x|$) witness w that certifies $(x,w) \in \mathcal{R}$, $x \in \mathcal{L}_{\mathcal{R}}$ can be decided in the polynomial time of $|x|$. Denote the interactive protocol between a (potentially unbounded) prover $\mathcal{P}$ and a PPT verifier $\mathcal{V}$ by $\langle \cdot, b \rangle \leftarrow \langle \mathcal{P}(\cdot,\cdot), \mathcal{V}(\cdot) \rangle$ where $b \in \{0,1\}$. Here, $b = 1$ indicates that $\mathcal{V}$ accepts the transcript of the interaction with $\mathcal{P}$; and $b = 0$ indicates $\mathcal{V}$ rejects. If an interactive protocol satisfies completeness, (perfect) zero-knowledge, and knowledge-soundness [15], we call it a ZKPoK protocol. It can be transferred into a non-interactive one by applying the Fiat-Shamir heuristic [16].

Definition 6 (Completeness). *An interactive protocol $\langle \mathcal{P}, \mathcal{V} \rangle$ for a relation $\mathcal{R}$ satisfies completeness, if for any $x \in \mathcal{L}_{\mathcal{R}}$ and w s.t., $(x,w) \in \mathcal{R}$:*

$$\Pr[\langle \cdot, 1 \rangle \leftarrow \langle \mathcal{P}(x,w), \mathcal{V}(x) \rangle] = 1.$$

Definition 7 ((Perfect) Zero-Knowledge). *An interactive protocol $\langle \mathcal{P}, \mathcal{V} \rangle$ for a relation $\mathcal{R}$ is (perfect) zero-knowledge if for any PPT adversary $\mathcal{A}$ there exists a PPT simulator $\mathcal{S}$ s.t. $\{\mathcal{S}^{\mathcal{A}}(x)\}_{x \in \mathcal{L}_{\mathcal{R}}} \approx \{\langle \mathcal{P}(x,w), \mathcal{A}(x) \rangle\}_{(x,w) \in \mathcal{R}}$ where $\langle \mathcal{P}(\cdot,\cdot), \mathcal{A}(\cdot) \rangle$ denotes the transcript of the interaction between $\mathcal{P}$ and $\mathcal{A}$, and "$\approx$" denotes (perfect) indistinguishability.*

Definition 8 (Knowledge-Soundness). *An interactive protocol $\langle \mathcal{P}, \mathcal{V} \rangle$ is proofs of knowledge (PoK) relative to an NP relation $\mathcal{R}$, if for any potentially unbounded adversarial prover $\mathcal{A}$ accepted by $\mathcal{V}$ on x, i.e., $\langle \cdot, 1 \rangle \leftarrow \langle \mathcal{A}(x), \mathcal{V}(x) \rangle$, with probability greater than ϵ, there exists a PPT knowledge extractor $\mathcal{K}^{\mathcal{A}}(x)$ (denoting that the extractor has rewinding black-box access to $\mathcal{A}$) that can output a value w satisfying $(x,w) \in \mathcal{R}$ with probability polynomial of ϵ.*

Following [9, Definition 2.2], the completeness, simulatability, and extraction properties of SoK are essentially reflected by the completeness, zero-knowledge, and knowledge-soundness above. We refer to the SoK as (non-interactive) proofs.

A secure public ledger. The formal security definitions from [20] are as follows.

Definition 12 (Persistence and Liveness of $(\Pi_{\mathsf{M}}, \mathcal{L})$)**.** Parameterized by $k, u \in \mathbb{N}$, a secure public ledger protocol satisfies:

- k-Persistence: If in a certain round, an honest user reports a ledger that contains a message m in a block more than k blocks away from the end of the ledger $\mathcal{L}$ (such messages will be called "stable"), then m will be reported by any honest user in the same position on $\mathcal{L}$, from this round on.
- (k,u)-Liveness: If a message m is given as input to all honest users continuously for u consecutive rounds, then all honest users will report this message more than k blocks from the end of $\mathcal{L}$, *i.e.*, all report it as stable.

B A Formal Treatment for DADB

This section presents syntax and security definitions of the DADB scheme. We also prove that our construction in Section 3.1 satisfies the security properties.

B.1 DADB Syntax

A DADB scheme consists of algorithms $(\mathsf{Setup}, \mathsf{Deposit}, \mathsf{dVrfy}, \mathsf{Bidding}, \mathsf{bVrfy})$.

- $\mathsf{Setup}(1^\kappa)$ takes as input the security parameter κ and outputs a public parameter pp *s.t.* $d \in \mathsf{pp}$ where d is the amount of deposit for one bid.
- $\mathsf{Deposit}(\mathsf{pp}, i)$ takes as input pp and a user index i. It outputs a deposit transaction d_i and a trapdoor td_i.
- $\mathsf{dVrfy}(\mathsf{pp}, i, \mathsf{d}_i)$ outputs 1 if d_i is valid; or 0 otherwise.
- $\mathsf{Bidding}(\mathsf{pp}, P, \mathsf{d}, \mathsf{td}, D)$ takes as input pp, a bidding string $P \in \{0,1\}^*$, a deposit transaction d and its trapdoor td, and a set of deposits D. It outputs a sealed bid $\mathsf{b} := (c_P, S, \pi)$ where c_P is the sealed bidding string; S and π are the serial number and proof indicating $\mathsf{d} \in D$.
- $\mathsf{bVrfy}(\mathsf{pp}, \mathsf{b}, D)$ outputs 1 if b is valid, *i.e.*, c_P is valid concerning P, and (S, π) is valid concerning D; or 0 otherwise.

Note that each bid should be associated with exactly one deposit transaction. Hence, we require S to be a unique value released during the issuance of a bid, designed to prevent any user from consuming the same deposit transaction twice.

B.2 DADB Security Definitions

The correctness, anonymity, and one-more bidding unforgeability for DADB are defined as follows.

Definition 9 (Correctness). *A DADB scheme is perfectly correct if the following properties hold for any $\kappa > 0$, any user i, and $(\mathsf{d}_i, \mathsf{td}_i) \leftarrow \mathsf{Deposit}(\mathsf{pp}, i)$.*

- *Deposit transaction verifies, i.e.,* $\mathsf{dVrfy}(\mathsf{pp}, i, \mathsf{d}_i) = 1$.
- *For any* $\mathsf{b} \leftarrow \mathsf{Bidding}(\mathsf{pp}, P, \mathsf{d}, \mathsf{td}, D')$ *s.t.* $\mathsf{d} \in D'$*, it verifies that* $\mathsf{bVrfy}(\mathsf{pp}, \mathsf{b}, D') = 1$. *Moreover, for any* $D \supseteq D'$*, we have* $\mathsf{bVrfy}(\mathsf{pp}, \mathsf{b}, D) = 1$.

Definition 10 (Anonymity). *A DADB scheme satisfies anonymity if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, given a user $i \in \{0,1\}$, the following probability is* $\mathsf{negl}(\kappa)$ *for any $\kappa > 0$ and $(\mathsf{d}_i, \mathsf{td}_i) \leftarrow \mathsf{Deposit}(\mathsf{pp}, i)$.*

$$\left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} (P, D, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{d}_0, \mathsf{d}_1); \\ b \xleftarrow{\$} \{0,1\}; \\ \mathsf{b}_b \leftarrow \mathsf{Bidding}(\mathsf{pp}, P, \mathsf{d}_b, \mathsf{td}, D \cup \{\mathsf{d}_0, \mathsf{d}_1\}); \\ b' \leftarrow \mathcal{A}_2(\mathsf{b}_b, \mathsf{st}) \end{array} \right] - \frac{1}{2} \right|$$

Definition 11 (One-More Bidding-Unforgeability). *A DADB scheme satisfies one-more bidding unforgeability if for any PPT adversary $\mathcal{A}$ that is given $n \leq \mathsf{poly}(\kappa)$ deposit transactions $(\mathsf{d}_i, \cdot) \leftarrow \mathsf{Deposit}(\mathsf{pp}, i)$. $\mathcal{A}$ can query, up to $q \leq \mathsf{poly}(\kappa)$ times, to a bidding oracle $\mathcal{O}_{\mathsf{Bidding}}(\cdot,\cdot,\cdot)$. The oracle works as follows: when queried (d_j, P_j, D) where $\{\mathsf{d}_i\}_{i\in[n]} \subseteq D$,*

- *returns $\perp$ if $\mathsf{d}_j \notin \{\mathsf{d}_i\}_{i\in[n]}$;*
- *otherwise, returns $\mathsf{b} \leftarrow \mathsf{Bidding}(\mathsf{pp}, P_j, \mathsf{d}_j, \mathsf{td}_j, D)$ to $\mathcal{A}$ and records (c_{P_j}, S_j) to a queried (message-serial number) set Q.*

The following probability is $\mathsf{negl}(\kappa)$ for any $\kappa > 0$.

$$\Pr\left[\begin{array}{l|l} \forall \mathsf{b}' \in \{\mathsf{b}'_i\}_{i\in[m+1]}, \{\mathsf{d}_i\}_{i\in[n]} \cup \{\mathsf{d}'_j\}_{j\in[m]} \subseteq D' : & \\ \mathsf{bVrfy}(\mathsf{pp}, \mathsf{b}', D') = 1 \wedge & \left(\{\mathsf{d}'_j\}_{j\in[m]}, \{\mathsf{b}'_i\}_{i\in[m+1]}\right) \\ \forall (c'_{P_i}, S'_i) \in \mathsf{b}'_i : & \quad \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Bidding}}}(\{\mathsf{d}_i\}_{i\in[n]}) \\ (c'_{P_i}, S'_i) \notin \mathsf{Q} \wedge S'_i \text{ is unique for each } \mathsf{b}'_i & \end{array}\right]$$

B.3 Proofs of Lemma 1

Lemma 1 (The Security of Deposit-and-Bid)**.** *Assuming a secure signature DS and a secure public ledger $\mathcal{L}$ (with u-liveness delay), our deposit-and-bid approach satisfies the following properties.*

- *Correctness (Definition 9).*
- *Anonymity (Definition 10) if the NITC scheme TC is CCA-hiding and the signature of knowledge proof π_{SoK} is at least computationally zero-knowledge.*
- *One-more bidding-unforgeability (Definition 11) if the NITC scheme TC is CCA-binding, the signature of knowledge proof π_{SoK} is at least computationally zero-knowledge and is knowledge-sound.*

The proof of correctness is straightforward. We show anonymity then.

Proof of Anonymity. Fix two users $i = 0, 1$. One first feeds the adversary with two deposit transactions $\mathsf{d}_0 = SIG_0(e, d, c_0) \leftarrow \mathsf{Deposit}(e, d, sk_0)$ and d_1 obtained similarly. As e and d are parameters that correspond to the secure ledger and are known to the public, we regard that the adversary knows c_0, pk_0 and c_1, pk_1 only. Due to the one-to-one correspondence between a deposit transaction and a committed serial number, we use the set of committed serial numbers instead of the set of deposit transactions. It is also sufficient for the adversary to distinguish committed serial numbers. Hence, public keys are omitted in the following analysis. The adversary chooses a set C' of committed serial numbers and a price P. On the uniformly randomly sampled $b \xleftarrow{\$} \{0, 1\}$, the adversary is offered with a bid $\mathsf{b}_b = (c_P, \pi_P, S_b, \pi_{b,\mathsf{SoK}}) \leftarrow \mathsf{Bidding}(P, S_b, r_b, C')$. The adversary is said to win the anonymity game if it outputs b' *s.t.* $b' = b$.

Within the challenge bids $\mathsf{b}_0, \mathsf{b}_1$, the commitment of price and the proof of the price range, c_P and π_P, are distributed identically. Hence, the adversary is

to work out b using the knowledge of:

$$c_0,\ c_1,\ S_b\ s.t.\ (c_b, \cdot) \leftarrow \mathsf{TC.Com}(S_b; r_b),\ \text{and}$$
$$\pi_{i,\mathsf{SoK}} \leftarrow \mathsf{SoK}[(c_P, \pi_P)]\{(c_i, r_i) : (c_i, \cdot) \leftarrow \mathsf{TC.Com}(S_i; r_i) \wedge c_i \in C' \cup \{c_0, c_1\}\}. \tag{B.1}$$

By (at least) computational zero-knowledge, with all but negligible probability, replacing the signature $\pi_{b,\mathsf{SoK}}$ with the signature $\pi_{b,\mathcal{S}}$ generated by the PPT simulator $\mathcal{S}$, which does not use r_b, has the same distribution as the origin one. Hence, except for a negligible probability, the adversary has an equal advantage in the anonymity game after replacing $\pi_{b,\mathsf{SoK}}$ with $\pi_{b,\mathcal{S}}$.

In Table 2, we show that if there exists $\mathcal{A}$ wins the anonymity game with probability $1/2+\epsilon$, we can construct an algorithm $\mathcal{B}$ which wins the CCA-hiding game with probability $1/2+\epsilon/4$. Notice that $b, b', s \in \{0,1\}$. There are eight cases

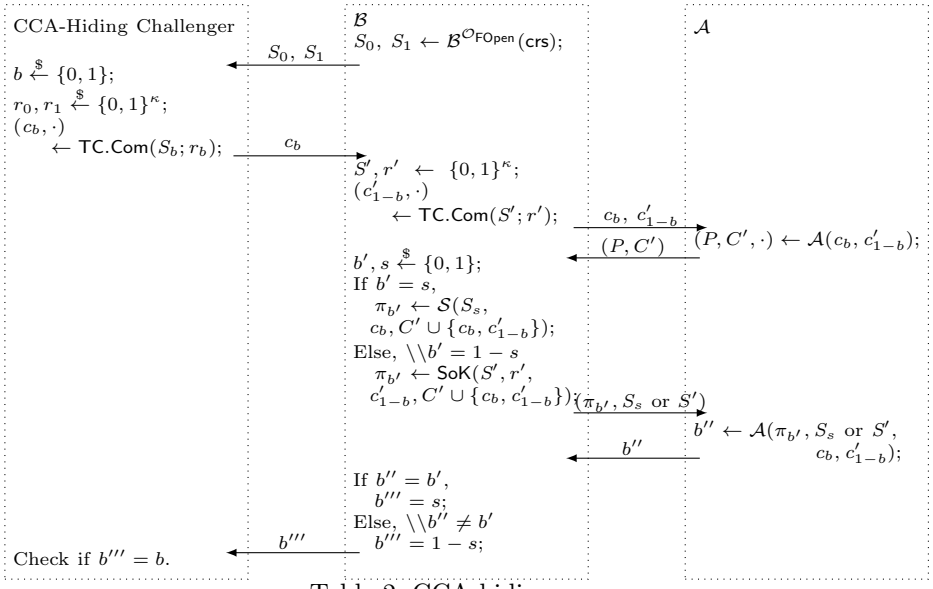


Table 2: CCA-hiding game

in all. We first analyze the case $s = 0$:

(1.1) If $(b, b') = (0, 0)$, then $(c_b, c'_{1-b}) = (c_0, c'_1)$ and $\pi_0 \leftarrow \mathcal{S}(S_0, c_0, C' \cup \{c_0, c'_1\})$. As the signature π_0 sent to $\mathcal{A}$ is valid, the adversary outputs $b'' = 0$ with probability $1/2 + \epsilon$. As $b'' = 0 = b'$ leads to $b''' = 0 = b$, the algorithm $\mathcal{B}$ wins with probability $1/2 + \epsilon$.

(1.2) If $(b, b') = (0, 1)$, then $(c_b, c'_{1-b}) = (c_0, c'_1)$ and $\pi_1 \leftarrow \mathsf{SoK}(S', r', c'_1, C' \cup \{c_0, c'_1\})$. As the signature π_1 sent to $\mathcal{A}$ is valid, the adversary outputs $b'' = 1$ with probability $1/2 + \epsilon$. As $b'' = 1 = b'$ leads to $b''' = 0 = b$, the algorithm $\mathcal{B}$ wins with probability $1/2 + \epsilon$.

(1.3) If $(b, b') = (1, 0)$, then $(c_b, c'_{1-b}) = (c_1, c'_0)$ and $\pi_0 \leftarrow \mathcal{S}(S_0, c_1, C' \cup \{c_1, c'_0\})$ which is an invalid signature. We will calculate the probability of $\mathcal{B}$ winning the game later.

(1.4) If $(b, b') = (1, 1)$, then only when $b'' = 0$ so that $b''' = 1$ can the algorithm $\mathcal{B}$ win the game. Now $(c_b, c'_{1-b}) = (c_1, c'_0)$ and $\pi_1 \leftarrow \mathsf{SoK}(S', r', c'_0, C' \cup \{c_1, c'_0\})$. As the signature π_1 sent to $\mathcal{A}$ is valid, with probability $1/2 - \epsilon$, the adversary outputs $b'' = 0$ and hence $\mathcal{B}$ wins.

Next, we sketch the analysis of the case $s = 1$:

(2.1) If $(b, b') = (0, 0)$, then $(c_b, c'_{1-b}) = (c_0, c'_1)$ and $\pi_0 \leftarrow \mathsf{SoK}(S', r', c'_1, C' \cup \{c_0, c'_1\})$. Only when $\mathcal{A}$ outputs $b'' = 1$ so that $b''' = 0$ can the algorithm $\mathcal{B}$ win the game. As the signature sent to $\mathcal{A}$ is valid, with probability $1/2 - \epsilon$, the adversary outputs $b'' = 1$, and hence $\mathcal{B}$ wins.

(2.2) If $(b, b') = (0, 1)$, then $(c_b, c'_{1-b}) = (c_0, c'_1)$ and $\pi_1 \leftarrow \mathcal{S}(S_1, c_0, C' \cup \{c_0, c'_1\})$ which is an invalid proof.

(2.3) If $(b, b') = (1, 0)$, then $(c_b, c'_{1-b}) = (c_1, c'_0)$ and $\pi_0 \leftarrow \mathsf{SoK}(S', r', c'_0, C' \cup \{c_1, c'_0\})$. The algorithm $\mathcal{B}$ wins with probability $1/2 + \epsilon$.

(2.4) If $(b, b') = (1, 1)$, then $(c_b, c'_{1-b}) = (c_1, c'_0)$ and $\pi_1 \leftarrow \mathcal{S}(S_1, c_1, C' \cup \{c_1, c'_0\})$. The algorithm $\mathcal{B}$ wins with probability $1/2 + \epsilon$.

As b, b', and s are randomly obtained, the probability that each of the cases (1.1), (1.2), (1.4), (2.1), (2.3), and (2.4) occurs with probability $1/8$. In these cases, the probabilities of the algorithm $\mathcal{B}$ winning the game are all known.

It suffices to work out $\mathcal{B}$'s winning probability when an invalid signature is sent to the adversary $\mathcal{A}$. Our goal is to show that this probability equals $1/2$. As b' is randomly obtained, it suffices to show that $\mathcal{A}$ can not judge $b' = 0$ or $= 1$ given an invalid signature. For this, we show that the (invalid) signatures obtained when $b' = 0$ or $= 1$ have the same distribution. If $b' = 0$, then the invalid signature is $\pi_0 \leftarrow \mathcal{S}(S_0, c_1, C' \cup \{c_1, c'_0\})$ in the case (1.3). If $b' = 1$, then the invalid signature is $\pi_1 \leftarrow \mathcal{S}(S_1, c_0, C' \cup \{c_0, c'_1\})$ in the case (2.2). As S_0 and S_1 are obtained in the same way, their distributions are the same. Notice that c_1 is obtained from $\mathsf{TC.Com}(S_1; r_1)$ and c_0 is obtained from $\mathsf{TC.Com}(S_0; r_0)$. Here r_1 and r_0 have the same distribution as they are obtained randomly. Hence c_1 and c_0 have the same distribution. These shows that π_0 and π_1 have the same distribution.

Overall, $\mathcal{B}$ wins with probability

$$2 \cdot \frac{1}{8} \cdot \left(\frac{1}{2} + \epsilon + \frac{1}{2} + \epsilon + \frac{1}{2} - \epsilon \right) + \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{2} + \frac{\epsilon}{4}.$$

□

Finally, we show one-more bidding-unforgeability and essentially follow [33, Appendix A.B].

Proof of One-more bidding-unforgeability. Let $\mathcal{A}$ be the adversary that wins the one-more bidding-unforgeability game. We construct an algorithm $\mathcal{B}$ that utilizes $\mathcal{A}$ to win the CCA-binding game:

– The input of the algorithm $\mathcal{B}$ is the common reference string crs of the commitment scheme TC so that $\mathcal{B}$ can use crs to prepare the input for $\mathcal{A}$.
– $\mathcal{B}$ samples $S_i, r_i \xleftarrow{\$} \{0,1\}^\kappa$ for $i \in [n]$. Using crs, $\mathcal{B}$ obtains c_i from $(c_i, \cdot) \leftarrow \mathsf{TC.Com}(\mathsf{crs}, S_i; r_i)$ for $i \in [n]$.
– $\mathcal{B}$ feeds $\mathcal{A}$ with $\{c_i\}_{i\in[n]}$. Then, $\mathcal{A}$ can generate arbitrary deposit transactions d_i for any $i \in [n]$. When $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{Bidding}}$ with (d_i, P_i, C) for $i \in [n]$ and $C \supseteq \{c_i\}_{i\in[n]}$, the algorithm $\mathcal{B}$ runs $\mathsf{Bidding}$ to obtain $\mathsf{b}_i = (m_i, S_i, \pi_{i,\mathsf{SoK}})$ and deliver it to $\mathcal{A}$, where $m_i = (c_{P_i}, \pi_{P_i})$. The algorithm $\mathcal{B}$ also records $\{(m_i, S_i)\}_{i\in\mathsf{Q}}$ queried by $\mathcal{A}$.
– From $\mathcal{A}$'s output[20], the algorithm $\mathcal{B}$ obtains m commitments $\{c'_j\}_{j\in[m]}$ and $m+1$ bids $\{\mathsf{b}'_j\}_{j\in[m+1]}$ with $\mathsf{b}'_j = (m'_j, S'_j, \pi'_{j,\mathsf{SoK}})$. Here, the signature of knowledge $\pi'_{j,\mathsf{SoK}}$ shows that some tuple (r^*_j, c^*_j) satisfies $(c^*_j, \cdot) \leftarrow \mathsf{TC.Com}(\mathsf{crs}, S'_j; r^*_j)$ and $c^*_j \in C'$ with $C' \coloneqq \{c_i\}_{i\in[n]} \cup \{c'_j\}_{j\in[m]}$.
– For the PPT SoK extractor (by knowledge-soundness), the algorithm $\mathcal{B}$ applies the extractor to $\pi'_{j,\mathsf{SoK}}$ for $j \in [m+1]$ and obtains $\{(r^*_j, c^*_j)\}_{j\in[m+1]}$.
– The algorithm $\mathcal{B}$ analyzes (r^*_j, c^*_j), $j \in [m+1]$ as below:
  (1) The extractor may fail[21]: (i) the extractor extracts nothing; (ii) $(c^*_j, \cdot) \neq \mathsf{TC.Com}(\mathsf{crs}, S'_j; r^*_j)$; (iii) $c^*_j \notin C'$.
  (2) Otherwise, we have $c^*_j \in C'$ for $j \in [m+1]$. In this case, we may have $c^*_j \in \{c_i\}_{i\in[n]}$. Assume so.
    (2.1) We may have $(S'_j, r^*_j) = (S_i, r_i)$ for some i and if $i \in \mathsf{Q}$, $m'_j \neq m_i$.
    (2.2) We may have $(S'_j, r^*_j) = (S_i, r_i)$ for some $i \in \mathsf{Q}$ such that $m'_j = m_i$.
    (2.3) We may have $(S'_j, r^*_j) \neq (S_i, r_i)$ for all i. The algorithm $\mathcal{B}$ sets
    $$(c, m, o, m', o') \coloneqq (c_i, S_i, r_i, S'_j, r^*_j).$$
  (3) In the remained case, we have $c^*_j \in \{c'_1, \cdots, c'_m\}$ for all $j \in [m+1]$. We have $c^*_i = c^*_j$ for some $i, j \in [m+1]$ with $i \neq j$. The algorithm $\mathcal{B}$ sets
    $$(c, m, o, m', o') \coloneqq (c^*_j, S'_i, r^*_i, S'_j, r^*_j).$$
– In the cases (1), (2.1), and (2.2), $\mathcal{B}$ outputs $\perp$. In the cases (2.3) and (3), output (c, m, o, m', o').

Notice that the occurrence of cases (2.3) and (3) violates the CCA-binding property[22]. It suffices to show that the probability of the case where (2.3) or (3) occurs is non-negligible.

[20] The adversary succeeding precisely means $\mathsf{bVrfy}(\mathsf{b}'_j, C') = 1$, (m'_j, S'_j) not recorded, and $S'_j \neq S'_i$ in b'_j and b'_i if $j \neq i$.

[21] The cases (ii) and (iii) mean that the obtained witness does not satisfy the set membership relation, i.e. violates the completeness of the ZKPoK.

[22] The occurrences of cases (2.3) or (3) violate the CCA-binding property. In the case (2.3), the inequality $(S'_j, r^*_j) \neq (S_i, r_i)$ for any i implies that the adversary successfully commits a bid using the deposit transaction provided by $\mathcal{B}$. In this case, the algorithm $\mathcal{B}$ finds for the commitment c_i two tuples (S'_j, r^*_j) and (S_i, r_i) of serial numbers and witnesses. This contradicts the CCA-binding property. If case (3) occurs, then the adversary successfully finds for a commitment in $\{c'_1, \ldots, c'_m\}$ two tuples of serial numbers and witnesses. This also contradicts the CCA-binding property.

As for (1), the probability of the extractor for each j failing is $\leq \mathsf{negl}(\kappa)$. We need the case where the extractor succeeds for all j. The successful probability is $> (1 - \mathsf{negl}(\kappa))^{m+1} > 1 - (m+1)\mathsf{negl}(\kappa)$.

Let $\mathcal{A}'$ denote the adversary induces the case (2.1). In this case, algorithm $\mathcal{B}$ behaves as below:

(1) Obtain $\widetilde{S}_i \xleftarrow{\$} \{0,1\}^\kappa$ such that $\widetilde{S}_i \neq S_i$.
(2) Rewind $\mathcal{A}'$ and feed it with $\{c_1, \ldots, c_n\}$.
(3) When $\mathcal{A}'$ queries $\mathcal{O}_{\mathsf{Bidding}}$ with (d_i, P_i, C), the algorithm $\mathcal{B}$ runs the simulator $\mathcal{S}$ provided by computational zero-knowledge to obtain a proof $\pi_{i,\mathcal{S}} \leftarrow \mathcal{S}(\widetilde{S}_i, c_i, C)$. Then $\mathcal{B}$ delivers the bid $\widetilde{\mathsf{b}}_i = (m_i, \widetilde{S}_i, \pi_{i,\mathcal{S}})$ to $\mathcal{A}'$.
(4) After receiving m commitments and $m+1$ bids, the algorithm $\mathcal{B}$ applies the PPT SoK extractor to the proofs in all $m+1$ bids to obtain the witnesses.

Let $\widetilde{r}_i$ denote the witness obtained from the extractor when it is applied to the bid $\mathsf{b}'_j = (m'_j, S'_j, \pi'_{j,\mathsf{SoK}})$ from $\mathcal{A}'$ satisfying $S'_j = \widetilde{S}_i$. Then the tuple $(c, m, o, m', o') \coloneqq (c_i, S_i, r_i, \widetilde{S}_i, \widetilde{r}_i)$ violates the CCA-binding property.

In the case (2.2), we have $(m'_j, S'_j) = (m_i, S_i)$ which has been delivered to the adversary. This implies that the adversary does not win in this case. This contradicts the assumption. In summary, as cases (1), (2.1), and (2.2) either occur with negligible probability or contradict the assumption, cases (2.3) and (3) occur with non-negligible probability. □

C Detailed Sidechain Protocol

C.1 Helper Functions

Committee selection. Given the hash function $H(\cdot)$, we denote the string of "0." concatenated with the hash value by $.H(\cdot) \coloneqq (0.H(\cdot))_2$. Let pp be the aforementioned parameter. On a high level, our committee selection is realized by applying the stake-based mechanism [10, Section 6] to the set of deposits $\bigcup_{i \in PK^e} D_i^e$ so that a user i is selected with a weight proportional to $|D_i^e|$. Following notations in loc. cit., the set of committees in round $r \geq 0$ step $s \geq 1$ is denoted by $SV^{r,s}$. Particularly, we call users in $SV^{r,1}$ potential leaders, and those in $SV^{r,s}$ for $s > 1$ verifiers.

Let $a_i \coloneqq |D_i|$ be the number of deposit transactions issued by user i in epoch e. To realize the committee selection, we propose two helper algorithms $\mathsf{GetVotes}$ and $\mathsf{GetMinHash}$ (Algorithm 1 and 2). In the algorithms, each user i *essentially* "gets a hash value as a lottery ticket" for each deposit transaction in D_i so that i has a_i hash values in each committee selection. In Algorithm 1, a deposit transaction becomes a vote if and only if the corresponding hash is small enough, which happens with probability $p^{r,s}$. In Algorithm 2, the minimal hash value among a_i hash values of a user's deposit transaction is obtained.

We abstract two helper algorithms below. For each user i, her committee credential of round r step s is $\sigma_i^{r,s} \coloneqq SIG_i(r, s, Q^{r-1})$.

– $\mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s})$ outputs the number of i's votes so that (1) i is a committee member if and only if $\mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s}) > 0$ and (2) i's power as a committee member is proportional to the number of her votes.
– $\mathsf{GetMinHash}(a_i, \sigma_i^{r,1})$ essentially outputs the "minimum hash" value *w.r.t.* $\sigma_i^{r,s}$.

Algorithm 1: GetVotes outputs the number of copies of i in $SV^{r,s}$.

function $\mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s})$;
Compute $y = .H(\sigma_i^{r,s})$;
`// Denote with` $\mathtt{p} \coloneqq p^{r,s}, \mathtt{a}_i \coloneqq a_i$ `in the following.`
for $x \in [a_i]_0$ **do**
  Compute $p_x = \binom{a_i}{x} p^x (1-p)^{a_i - x}$;
  if $y \in [0, p_0]$ **then**
    Return 0 ;
  else
    if $y \in \left(\sum_{x'<x} p_{i,x'}, \sum_{x' \le x} p_{i,x'}\right]$ **then**
      Return x ;
    end
  end
end

Algorithm 2: GetMinHash outputs the "minimum hash" *w.r.t.* $\sigma_i^{r,s}$.

function $\mathsf{GetMinHash}(a_i, \sigma_i^{r,1})$;
if $\mathsf{GetVotes}(p^{r,1}, a_i, \sigma_i^{r,1}) = 0$ **then**
  Return 0 ;
end
Compute $y = .H(\sigma_i^{r,s})$;
`// Denote with` $\mathtt{a}_i \coloneqq a_i$ `in the following.`
for $x \in [2^\kappa]_0$ **do**
  Compute $p_{i,x} = \left(\frac{2^\kappa - x + 1}{2^\kappa + 1}\right)^{a_i} - \left(\frac{2^\kappa - x}{2^\kappa + 1}\right)^{a_i}$;
  if $y \in [0, p_0]$ **then**
    Return 0 ;
  else
    if $y \in \left(\sum_{x'<x} p_{i,x'}, \sum_{x' \le x} p_{i,x'}\right]$ **then**
      Return x ;
    end
  end
end

Message verification. VrfyMsg for verifying messages is in Algorithm 3.

Algorithm 3: VrfyMsg verifies messages of step $s \geq 1$.

function $\mathsf{VrfyMsg}(p^{r,s}, a_i, \mathcal{C}^{r-1}, m_i^{r,s})$;
Parse $\sigma_i^{r,s} \in m_i^{r,s}$;
if $\mathsf{SigVrfy}(pk_i, \sigma_i^{r,s}) = 0 \vee \mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s}) = 0$ **then**
  Return 0 ;
end
if $s = 2$ **then**
  Parse $m_i^{r,2} = (ESIG_i(v_i), \sigma_i^{r,2}, SIG_\ell(Q^{r-1}))$;
  if $\mathsf{SigVrfy}(pk_i^{r,2}, ESIG_i(v_i)) = 0 \wedge \mathsf{SigVrfy}(pk_\ell, SIG_\ell(Q^{r-1})) = 0$ **then**
    Return 0 ;
  end
else if $s = 3$ **then**
  Parse $m_i^{r,3} = (ESIG_i(v_i), \sigma_i^{r,3})$;
  if $\mathsf{SigVrfy}(pk_i^{r,3}, ESIG_i(v_i)) = 0$ **then**
    Return 0 ;
  end
else if $s > 4 \wedge s \in \mathbb{N}$ **then**
  Parse $m_i^{r,s} = (ESIG_i(b_i), ESIG_i(v_i), \sigma_i^{r,s})$;
  if $\mathsf{SigVrfy}(pk_i^{r,s}, ESIG_i(b_i)) = 0 \vee \mathsf{SigVrfy}(pk_i^{r,s}, ESIG_i(v_i)) = 0$ **then**
    Return 0 ;
  end
else if $s = 1$ **then**
  Parse $m_i^{r,1} = (B_i^r, esig_i(H(B_i^r)), \sigma_i^{r,1})$;
  if $\mathsf{SigVrfy}(pk_i^{r,1}, H(B_i^r), esig_i(H(B_i^r))) = 0$ **then**
    Return 0 ;
  end
  Parse $\mathcal{C}^{r-1} = B^0 || \ldots || B^{r-1}$ and $\mathsf{B}^{r'} \in B^{r'}$ for all $r' \in [r-1]_0$;
  Parse $B_i^r = (r_i, H_i, SIG_i(Q_i^{r-1}), \mathsf{B}_i^r)$;
  if $r_i \neq r \vee H_i \neq H(B^{r-1}) \vee Q_i^{r-1} \neq H(SIG_{\ell^{r-1}}(Q^{r-2}), r)$
   $\vee \mathsf{SigVrfy}(pk_i, SIG_i(Q_i^{r-1})) = 0$ **then**
    Return 0 ;
  end
  for $\mathsf{b} \in \mathsf{B}_i^r$ **do**
    if $\mathsf{bVrfy}(\mathsf{b}, C) = 0$ **then**
      Return 0 ;
    else
      Parse $\mathsf{b} = (c_P, \pi_P, S, \pi_{\mathsf{SoK}})$;
      if $\exists \mathsf{b}' \in \mathsf{B}_i^r \cup \left(\bigcup_{r' \in [r-1]_0} \mathsf{B}^{r'}\right)$ *s.t.* $S \in \mathsf{b}' \wedge \mathsf{b}' \neq \mathsf{b}$ **then**
        Return 0 ;
      end
    end
  end
else
  Return 0 ;
end
Return 1 ;

C.2 Our Leader Selection with Block Qualification

The specification of $\mathsf{SelectL}$ is presented in Algorithm 4.

Algorithm 4: $\mathsf{SelectL}$ outputs the selected leader index of round r.

function $\mathsf{SelectL}(\mathsf{MP}_i^r, M_i^{r,1})$;
  `// Extract the set of serial numbers from the mempool.`
$\mathsf{SP}_i^r = \emptyset$;
for $\mathsf{b} \in \mathsf{MP}_i^r$ **do**
  Parse $\mathsf{b} = (c_P, \pi_P, S, \pi_{\mathsf{SoK}})$;
  $\mathsf{SP}_j^r = \mathsf{SP}_j^r \cup \{S\}$;
end
  `// Extract the set of serial numbers from each candidate block.`
$V = \emptyset$;
for $m_j^{r,1} \in M_i^{r,1}$ **do**
  Parse $m_j^{r,1} = (B_j^r, esig_j(H(B_j^r)), \sigma_j^{r,1})$ with
    $B_j^r = (r, H(B^{r-1}), SIG_j(Q^{r-1}), \mathsf{B}_j^r)$;
  $\mathsf{S}_j^r = \emptyset$;
  for $\mathsf{b} \in \mathsf{B}_j^r$ **do**
    Parse $\mathsf{b} = (c_P, \pi_P, S, \pi_{\mathsf{SoK}})$;
    $\mathsf{S}_j^r = \mathsf{S}_j^r \cup \{S\}$;
  end
  $V = V \cup \{\mathsf{S}_j^r\}$;
end
Compute $y = \max_{\mathsf{S}_j^r \in V}(|\mathsf{S}_j^r \cap \mathsf{SP}_i^r|)$;
  `// Leader selection with block qualification.`
$W = \emptyset$;
if $y \geq L$ **then**
  for $\mathsf{S}_j^r \in V$ **do**
    if $|\mathsf{S}_j^r \cap \mathsf{SP}_i^r| \geq \mathfrak{h} y$ **then**
      $W = W \cup \{j\}$;
    end
  end
else
  $W = \{j \mid \mathsf{S}_j^r \in V\}$
end
Return $\arg\min_{j \in W} \mathsf{GetMinHash}(a_j, \sigma_j^{r,1})$;

C.3 Step $s \geq 3$ in The Sidechain Protocol

These steps are almost identical to the Algorand protocol [10], with the committee selection modified to $\mathsf{GetVotes}$, and the leader reelection modified to use $\mathsf{GetMinHash}$ instead of normal hash functions.

Step 3: The Second Step of GC

In any round $r \geq 0$, each user i starts her step 3 as soon as she finishes her step 2. The user waits[a] a maximum amount of time 2λ. She performs as follows *during* the waiting period.

1. If i has received $\geq t_H$ votes embedding the same value v, *i.e.*, $\sum_{j \in W} x_j \geq t_H$ with

$$x_j \coloneqq \mathsf{GetVotes}(p^{r,2}, a_j, \sigma_j^{r,2}) \text{ and}$$
$$W \coloneqq \left\{ j \;\middle|\; (ESIG_j(v), \sigma_j^{r,2}) \in M_i^{r,2} \right\},$$

   she stops waiting and sets[b] $v_i \coloneqq v$. Otherwise, when t_3 runs out, she sets $v_i \coloneqq \perp$.
2. She runs $\mathsf{GetVotes}(p^{r,3}, a_i, \sigma_i^{r,3}) = x$. She stops and propagates nothing if $x = 0$; Otherwise, she prepares a message $m_i^{r,3} \coloneqq (ESIG_i(v_i), \sigma_i^{r,3})$ with her ephemeral key pair $(sk_i^{r,3}, pk_i^{r,3})$, destroys $sk_i^{r,3}$, and propagates $m_i^{r,3}$.

[a] Hence the maximal total waiting time since the user starts her round r is $t_3 \coloneqq t_2 + 2\lambda = 3\lambda + 2\Lambda$.

[b] If the user has received two valid messages from a user j, *e.g.*, $(ESIG_j(v), \sigma_j^{r,2})$ and $(ESIG_j(v'), \sigma_j^{r,2})$ *s.t.* $v \neq v'$, they are counted for v and v', respectively. Though, this occurs only when j is malicious.

Step 4: The Output of GC and The First Step of BBA⋆

In any round $r \geq 0$, each user i starts her step 4 as soon as she finishes her step 3. The user waits[a] a maximum amount of time 2λ. She performs as follows *during* the waiting period.

1. The user computes the output of GC with:
   - If i has received at least t_H votes embedding the same $v \neq \perp$, she stops waiting and sets $v_i \coloneqq v$ and $g_i \coloneqq 2$.
   - If i has received at least t_H votes embedding $\perp$, she stops waiting and sets $v_i \coloneqq \perp$ and $g_i \coloneqq 0$.
   - Otherwise, when 2λ time runs out:
     - If there exists a value $v \neq \perp$ *s.t.* the user has received at least $\lceil \frac{t_H}{2} \rceil$ votes embedding v, then $v_i \coloneqq v$ and $g_i \coloneqq 1$;
     - Otherwise, she sets $v_i \coloneqq \perp$ and $g_i \coloneqq 0$.
2. Once (v_i, g_i) is set, the user computes the input of BBA⋆ with: $b_i = 0$ if $g_i = 2$, and $b_i = 1$ otherwise.
3. She runs $\mathsf{GetVotes}(p^{r,4}, a_i, \sigma_i^{r,4}) = x$. She stops and propagates nothing if $x = 0$; Otherwise, she prepares a message $m_i^{r,4} \coloneqq (ESIG_i(b_i), ESIG_i(v_i), \sigma_i^{r,4})$ with her ephemeral key pair $(sk_i^{r,4}, pk_i^{r,4})$, destroys $sk_i^{r,4}$, and propagates $m_i^{r,4}$.

[a] Hence, the maximum total waiting time since the user starts her round r is $t_4 := t_3 + 2\lambda = 5\lambda + 2\Lambda$.

Step $s \geq 5, s - 2 \equiv 0 \mod 3$: Coin-Fixed-To-0 in BBA⋆

In any round $r \geq 0$, each user i starts her step s as soon as she finishes her step $s-1$. The user waits[a] a maximum amount of time 2λ. She performs as follows *during* the waiting period.

- *Ending Condition* 0: If at any point there exists a string $v \neq \perp$ and a step s' *s.t.*:
  1. $s' \in [5 \,..\, s]$ is a coin-fixed-to-0 step, *i.e.*, $s' - 2 \equiv 0 \mod 3$;
  2. i has received at least t_H votes embedding the same $(0, v)$;
  3. Parse $v = (H_\ell, \ell)$, i has received a valid step 1 message $m_\ell^{r,1} = (B_\ell^r, esig_\ell(H(B_\ell^r)), \sigma_\ell^{r,1})$, *s.t.* $H_\ell = H(B_\ell^r)$.

  Then, i stops waiting and ends her round r immediately. She sets[b] her $CERT^r$ as the set of messages $m_j^{r,s'-1}$ together with $SIG_\ell(Q^{r-1})$.
- *Ending Condition* 1: If at any point there exists a step s' *s.t.*:
  1. $s' \in [6 \,..\, s]$ is a coin-fixed-to-1 step, *i.e.*, $s' - 2 \equiv 1 \mod 3$;
  2. i has received at least t_H votes embedding 1, *i.e.*, $\sum_{j \in W} x_j \geq t_H$ with:[c]

$$x_j := \mathsf{GetVotes}(p^{r,s'-1}, a_j, \sigma_j^{r,s'-1}) \text{ and}$$
$$W := \left\{ j \;\middle|\; (ESIG_j(1), ESIG_j(v_j), \sigma_j^{r,s'-1}) \in M_i^{r,s'-1} \right\}.$$

  Then, i stops waiting and ends her round r immediately. She sets $B^r = B_\epsilon^r$, and sets $CERT^r$ as the set of messages $m_j^{r,s'-1}$ together with $H(Q^{r-1}, r)$.
- If at any point i has received at least t_H votes embedding 1, she stops waiting and sets $b_i := 1$; Otherwise, when time runs out, i sets $b_i := 0$, *i.e.*, coin fixed to 0.
- She runs $\mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s}) = x$. She propagates nothing if $x = 0$; Otherwise, she prepares a message $m_i^{r,s} := (ESIG_i(b_i), ESIG_i(v_i), \sigma_i^{r,s})$, where v_i is computed from step 4, with her ephemeral key pair $(sk_i^{r,s}, pk_i^{r,s})$, destroys $sk_i^{r,s}$, and propagates $m_i^{r,s}$.

[a] Hence, the maximum total waiting time since the user starts her round r is $t_s := t_{s-1} + 2\lambda = (2s-3)\lambda + 2\Lambda$. The equality $t_s = (2s-3)\lambda + 2\Lambda$ holds for $s \geq 5$.

[b] Users only include $SIG_\ell(Q^{r-1})$ in their $CERT^r$ instead of the block head in Algorand [10].

[c] v_j's are not required to be identical in this case.

Step $s \geq 6, s-2 \equiv 1 \mod 3$: Coin-Fixed-To-1 in $\mathsf{BBA}^\star$

In any round $r \geq 0$, each user i starts her step s as soon as she finishes her step $s-1$. The user waits a maximum amount of time 2λ. She performs as follows *during* the waiting period.

- *Ending Conditions* 0 *and* 1: The same instructions as in coin-fixed-to-0 steps.
- If at any point i has received at least t_H votes embedding 0, *i.e.*, $\sum_{j \in W} x_j \geq t_H$ with

$$x_j \coloneqq \mathsf{GetVotes}(p^{r,s-1}, a_j, \sigma_j^{r,s-1}) \text{ and}$$
$$W \coloneqq \left\{ j \;\middle|\; (ESIG_j(0), ESIG_j(v_j), \sigma_j^{r,s-1}) \in M_i^{r,s-1} \right\},$$

  she stops waiting and sets $b_i \coloneqq 0$; Otherwise, when time runs out, i sets $b_i \coloneqq 1$, *i.e.*, coin fixed to 1.
- She runs $\mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s}) = x$. She propagates nothing if $x = 0$; Otherwise, she prepares a message $m_i^{r,s} \coloneqq (ESIG_i(b_i), ESIG_i(v_i), \sigma_i^{r,s})$, where v_i is computed from step 4, with her ephemeral key pair $(sk_i^{r,s}, pk_i^{r,s})$, destroys $sk_i^{r,s}$, and propagates $m_i^{r,s}$.

Step $s \geq 7, s-2 \equiv 2 \mod 3$: Coin-Genuinely-Flipped in $\mathsf{BBA}^\star$

In any round $r \geq 0$, each user i starts her step s as soon as she finishes her step $s-1$. The user waits a maximum amount of time 2λ. She performs as follows *during* the waiting period.

- *Ending Conditions* 0 *and* 1: The same instructions as in coin-fixed-to-0 steps.
- If at any point i has received at least t_H votes embedding 0, she stops waiting and sets $b_i \coloneqq 0$.
- If at any point i has received at least t_H votes embedding 1, she stops waiting and sets $b_i \coloneqq 1$.
- Otherwise, when time runs out, denote the committee from whom i has received valid messages by $SV_i^{r,s-1}$, i selects with:

$$\ell = \underset{j \in SV_i^{r,s-1}}{\arg\min}\, \mathsf{GetMinHash}(\sigma_j^{r,s-1})$$

  , and sets $b_i \coloneqq \mathsf{lsb}(H(\sigma_j^{r,s-1}, r))$, *i.e.*, if i has not received enough signatures for 0 or 1, she flips the universally available coin (the hash function, modeled as a random oracle) to decide her b_i.
- She runs $\mathsf{GetVotes}(p^{r,s}, a_i, \sigma_i^{r,s}) = x$. She propagates nothing if $x = 0$; Otherwise, she prepares a message $m_i^{r,s} \coloneqq (ESIG_i(b_i), ESIG_i(v_i), \sigma_i^{r,s})$, where

v_i is computed from step 4, with her ephemeral key pair $(sk_i^{r,s}, pk_i^{r,s})$, destroys $sk_i^{r,s}$, and propagates $m_i^{r,s}$.

D Configuration Options and Potential Extension

In this subsection, we propose a protocol where waiting for Λ in step 1 is not needed. The difference between the protocol in this section and the one in Section 3.2 is mainly step 1 and the algorithm that is used in step 2 for verifying messages. Moreover, we have $t_s = (2s-3)\lambda + \Lambda$ for $s \geq 2$ for the maximal total waiting time since a user starts her round r. We only list the alternative step 1 and the algorithm that used step 2.

Step 1: Block Proposal

In any round $r \geq 0$ of Π^{pp}, each user i starts her (own) step 1 as soon as she starts her round r. The user performs as follows then.

1. i gets her mempool MP_i^r by collecting bids b from the network[a] *s.t.*:
   (1) $\mathsf{bVrfy}(\mathsf{b}, C) = 1$;
   (2) $\nexists \mathsf{b}' \in \mathsf{MP}_i^r \cup \left(\bigcup_{r' \in [r-1]_0} \mathsf{B}^{r'}\right)$ satisfying $\mathsf{b}' \neq \mathsf{b}$ and the serial number S in both b and b'.
2. i runs $\mathsf{GetVotes}(p^{r,1}, a_i, \sigma_i^{r,1}) = x$. She ends her step 1 if $x = 0$ (*i.e.*, $i \notin SV^{r,1}$); Otherwise, she performs as follows.
   i sets $\mathsf{B}_i^r = \mathsf{MP}_i^r$, computes her candidate block $B_i^r = (r, H(B^{r-1}), SIG_i(Q^{r-1}), \mathsf{B}_i^r)$, prepares a message $m_i^{r,1} := (B_i^r, esig_i(H(B_i^r)), \sigma_i^{r,1})$ with her ephemeral key pair $(sk_i^{r,1}, pk_i^{r,1})$, destroys $sk_i^{r,1}$, and propagates $m_i^{r,1}$.[b]

[a] Some of these bids may come from candidate blocks in previous rounds.

[b] Note that, unlike Algorand [10], users in our protocol only propagate one message in step 1, as each verifier in step 2 must receive complete blocks to select a leader in her view.

As for the alternated step 2, we need an alternated $\mathsf{VrfyMsg}$ in Algorithm 5 to verify messages from the alternated step 1. We only specified the part that is different from Algorithm 3. Let $M_i^{r,1}$ denote the set of all valid step 1 messages collected by user i using the alternated $\mathsf{VrfyMsg}$.

Finally, we list two lemmas for the alternated protocol that are noticeably different from those for Aucrand. Further security analysis remains one of our future goals.

Lemma 13. *Assume that all honest users are sure about the same B^{r-2} and the same B^{r-1} in respectively Γ^{r-1} and Γ^r. All honest users receive the block B^{r-2} before start their round r.*

Algorithm 5: Alternative $\mathsf{VrfyMsg}$ for configuration.

function $\mathsf{VrfyMsg}(p^{r,s}, a_i, \mathcal{C}^{r-2}, m_i^{r,s})$;
· · · · · ·
else if $s = 1$ **then**
  Parse $m_i^{r,1} = (B_i^r, esig_i(H(B_i^r)), \sigma_i^{r,1})$;
  if $\mathsf{SigVrfy}(pk_i^{r,1}, H(B_i^r), esig_i(H(B_i^r))) = 0$ **then**
    Return 0 ;
  end
  Parse $\mathcal{C}^{r-2} = B^0||\dots||B^{r-2}$ and $\mathsf{B}^{r'} \in B^{r'}$ for all $r' \in [r-2]_0$;
  Parse $B_i^r = (r_i, H_i, SIG_i(Q_i^{r-1}), \mathsf{B}_i^r)$;
  if $r_i \neq r \vee H_i \neq H(B^{r-1}) \vee Q_i^{r-1} \neq H(SIG_{\ell^{r-1}}(Q^{r-2}), r)$
   $\vee\, \mathsf{SigVrfy}(pk_i, SIG_i(Q_i^{r-1})) = 0$ **then**
    Return 0 ;
  end
  for $\mathsf{b} \in \mathsf{B}_i^r$ **do**
    if $\mathsf{bVrfy}(\mathsf{b}, C) = 0$ **then**
      Return 0 ;
    else
      Parse $\mathsf{b} = (c_P, \pi_P, S, \pi_{\mathsf{SoK}})$;
      if $\exists \mathsf{b}' \in \mathsf{B}_i^r \cup \left(\bigcup_{r' \in [r-2]_0} \mathsf{B}^{r'}\right)$ *s.t.* $S \in \mathsf{b}' \wedge \mathsf{b}' \neq \mathsf{b}$ **then**
        Return 0 ;
      end
    end
  end
else
  Return 0 ;
end
Return 1 ;

Proposition 14. *Assume that all honest users are sure about the same* $B^{r'-1}$ *in the time interval* $I^{r'}$ *for* $r' \in [r, r+10]$*. If* $L \geq |C| \cdot 5\%$*, where* C *denotes the set of all serial numbers, then the bad case can not consecutively happen for* $r' \in [r..r+10]$.[23]

In fact, following the proof of Proposition 4(3), one may show that: If the bad case consecutively in rounds $r' \geq r$, then there is $\geq \left(\left(\frac{1}{\mathfrak{h}}\right)^{\lfloor r'-r \rfloor/2} - \mathfrak{h}\right) L$ serial numbers in $\mathsf{B}^{r'}$ are owned by the adversary.

E Details on Security Analysis

This section is devoted to collecting the details of results in Section 4.

[23] In this case, slightly reducing the value $\mathfrak{h}$ could result in a prominent decrease of the consecutive rounds length (similar to Footnote 10). For example, if $\mathfrak{h} = 1/2$, then the bad case can not consecutively happen for $r' \in [r..r+6]$.

E.1 Proof of Lemmas 3 and Proposition 4

Let $\alpha_i^{r,s}$ and $\beta_i^{r,s}$ denote respectively the (local) time a user i starts and ends her round r step s for integers $r \geq 0$ and $s \geq 1$.

Lemma 3. (1) *Assume that all honest users are sure about the same B^{r-1}. Let v denote the value in the message signed by verifiers in $SV^{r,s}$ If there is a value v getting $\geq t_H$ votes from verifiers in $SV^{r,s}$, then there exists* no *other value $v' \neq v$ such that v' and v have the same length and v' gets $\geq t_H$ votes from verifiers in $SV^{r,s}$.*

(2) *Assume that all honest users are sure about the same B^{r-1} in the time interval I^r and the same B^{r-2}. Any honest user i receive the block B^{r-1} before collecting her mempool MP_i^r of round r.*

Proof. (1) For each step $s = 2, 3$ of round r, the value v equals $(H(B_\ell^r), \ell)$. For $s \geq 4$, this value v may be equal to b, $\big(H(B_\ell^r), \ell\big)$, or $\big(b, H(B_\ell^r), \ell\big)$ for $b = 0, 1$. One can show this lemma using Eq. 4.1 following [10, Lemma 5.7] and the proof is omitted.

(2) It suffices to show that all honest users receive the block B^{r-1} before waiting time Λ ends in round r step 1. For the case where no value v from the messages sent by $SV^{r-1,2}$ gets $\geq t_H$ votes, we know that $v = \perp$ for all verifiers in $HSV^{r-1,s}$ where $s \geq 3$. Hence, $b = 1$ for all verifiers in $HSV^{r-1,4}$. In step 5, as $|HSV^{r-1,4}| > t_H$, all verifiers in $HSV^{r-1,5}$ sets $b = 1$. By Lemma 3(1), the value $b = 0$ gets $< t_H$ votes, and hence, no user is able to set up $CERT^{r-1}$, *i.e.*, no user can trigger Ending Condition 0 and finish round $r-1$ in step 5. In step 6, as $|HSV^{r-1,5}| > t_H$, each honest user is able to collect $\geq t_H$ votes for $b = 1$ and set up $CERT^{r-1}$. In this case, the block $B^r = B_\epsilon^r$ is empty and the result follows immediately.

Let $v = (H(B_\ell^{r-1}), \ell)$ be the value which gets $\geq t_H$ votes from the messages of $SV^{r-1,2}$. As $|MSV^{r-1,2}| < t_H$ (*i.e.*, the malicious verifiers in $MSV^{r-1,2}$ own $< t_H$ votes), there exists a verifier $i \in HSV^{r-1,2}$ who has propagated a valid message embedding v. As i is honest, she has received and checked B_ℓ^{r-1}. She also helps propagate B_ℓ^{r-1} regardless ℓ is honest or malicious. Her propagation reaches all users no later than Λ after the end of round $r-1$, *i.e.*, no later than $\Lambda + T^r$. Notice $\alpha_{i'}^{r,1} \geq T^r$ for any honest user i'. The user i' is able to receive B_ℓ^{r-1} before $\alpha_{i'}^{r,1} + \Lambda$, as desired. □

Proposition 4. (1) *If the good case happens and some honest potential leader ℓ has the minimal hash, then ℓ is the r-leader.*

(2) *Put $\mathsf{SP}_H^r \coloneqq \bigcup_i \mathsf{SP}_i^r$, where the union extends over verifiers in $HSV^{r,2}$. If $\geq \mathfrak{h}$ of serial numbers in SP_H^r are known by verifiers in $HSV^{r,1} \cup HSV^{r,2}$, i.e., $\geq \mathfrak{h}$ of serial numbers in SP_H^r belong to $\mathsf{SP}_{i'}^r$ for any verifier $i' \in HSV^{r,1} \cup HSV^{r,2}$, then the good case happens.*

(3) *Admit $\mathfrak{h} = 2/3$. Assume that all honest users are sure about the same $B^{r'-1}$ in the time interval $I^{r'}$ for $r' \in [r, r+5]$. If $L \geq |C| \cdot 5\%$, where C denotes the set of all serial numbers, then the bad case can not consecutively happen for $r' \in [r..r+5]$.*

Proof of Proposition 4(1). As the good case happens, the honest potential leader ℓ belongs to W_i for each verifier $i \in HSV^{r,2}$. As ℓ has the minimal hash value, ℓ is the leader for each verifier $i \in HSV^{r,2}$. Since $|HSV^{r,2}| > t_H$, the potential leader ℓ is the r-leader. □

Proof of Proposition 4(2). For $i' \in HSV^{r,1}$ and $i \in HSV^{r,2}$, we need to show $i' \in W_i$. The assumption implies that $|\mathsf{SP}^r_{i'} \cap \mathsf{SP}^r_i| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_H|$. As $\mathsf{S}^r_{i'} = \mathsf{SP}^r_{i'}$, we know $|\mathsf{S}^r_{i'} \cap \mathsf{SP}^r_i| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_H|$. For $y = \max_{V_i}\{|\mathsf{S}^r_j \cap \mathsf{SP}^r_i|\}$, we know $y < |\mathsf{SP}^r_i|$. By hypothesis, we have

$$|\mathsf{S}^r_{i'} \cap \mathsf{SP}^r_i| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_H| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_i| > \mathfrak{h}y,$$

as desired. □

Proof of Proposition 4(3). If i (the verifier i in the definition of the bad case) disqualifies the candidate block of i', she has received a message from $SV^{r,1}$ embedding the set of bids B^r that can be verified by $\mathsf{VrfyMsg}$ such that $|\mathsf{B}^r| \geq L$ and $\geq (1-\mathfrak{h})L$ serial numbers in B^r are in SP^r_i but not in $\mathsf{SP}^r_{i'}$ (cf. the inverse of Proposition 4(2)). As honest users propagate their bids, $\geq (1-\mathfrak{h})L$ serial numbers in B^r are owned by the adversary due to the one-more bidding-unforgeability. Note that the honest verifier i helps propagate B^r in her round r step 2.[24] Similar to Lemma 3(2), each honest user i' received the bids in B^r before they start to collect her serial number pool $\mathsf{SP}^{r+1}_{i'}$.

In round $r+1$, $\mathsf{SP}^{r+1}_{i'}$ contains serial numbers in B^r by the protocol. If the bad case happens in round $r+1$, a verifier in $HSV^{r+1,2}$, say i, has received a message embedding B^{r+1} that can be verified by $\mathsf{VrfyMsg}$ such that $|\mathsf{B}^{r+1}| \geq \frac{1}{\mathfrak{h}}|\mathsf{B}^r| \geq \frac{1}{\mathfrak{h}}L$ and $\geq \left(\frac{1}{\mathfrak{h}} - \mathfrak{h}\right)L$ serial numbers in B^{r+1} are in SP^{r+1}_i but not in $\mathsf{SP}^{r+1}_{i'}$. Hence, $\geq \left(\frac{1}{\mathfrak{h}} - \mathfrak{h}\right)L$ serial numbers in B^{r+1} are owned by the adversary.

By induction, if the bad case consecutively occurs in rounds $r' \geq r$, then there are $\geq \left(\left(\frac{1}{\mathfrak{h}}\right)^{r'-r} - \mathfrak{h}\right)L$ serial numbers in $\mathsf{B}^{r'}$ are owned by the adversary. With $\mathfrak{h} = \frac{2}{3}$, we have $\left(\left(\frac{3}{2}\right)^5 - \frac{2}{3}\right)L \approx |C| \cdot 34.6\%$. This is impossible by HMS. □

Remark 15. If the adversary lets no malicious user become the leader in the views of $\geq t_H - |MSV^{r,2}|$ verifiers in $HSV^{r,2}$, then no malicious leader will be regarded as the leader by any honest user in step $s \geq 3$. Notice that the adversary can not predict $HSV^{r,s}$ for $s \geq 2$ and $|HSV^{r,2}|/2 + |MSV^{r,2}| < t_H$ (by Eq. 4.1). By the intersection in Eq. 3.1 (see also Algorthim 4), to let some malicious user become the leader for $\geq t_H - |MSV^{r,2}|$ verifiers in $HSV^{r,2}$, the adversary must send its bids to $> (t_H - |MSV^{r,2}|)/|HSV^{r,2}| > 1/2$ of honest users before they collect their serial number pool. If so, in the leader selection, malicious users must compete with the hash values of $> 1/2$ honest potential leaders who know the adversarial bids.

[24] A verifier i in $HSV^{r,2}$ not only helps propagate the block of the leader (in i's view) but also collects (as in step 1) and helps propagate those bids which are not in the leader's block but in the received candidate blocks.

E.2 The Main Theorem for the Security of Aucrand

Recall the main theorem of Algorand [10, Theorem 1] is proved under the honest majority of user assumption (HMU). After replacing HMU with the honest majority of money assumption (HMM), the main theorem of Algorand with HMM can be proved by treating each unit of money as an individual user (See [10, Section 6]). As the committee selection is based on the number of deposit transactions $a_i = |D_i|$ issued by each user $i \in PK$, HMD plays a similar role as HMM. It turns out that the proof of the main theorem for Aucrand with HMD is similar to that of the main theorem of Algorand. Table 3 exhibits the relations between assumptions.

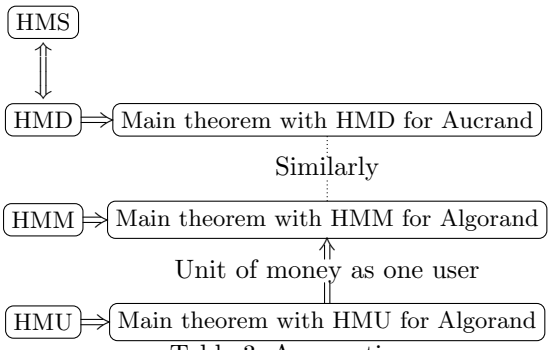


Table 3: Assumptions

Theorem 1. *The following properties hold with overwhelming probability for each round* $r \geq 0$:

(1) (a) *All honest users agree on the same block* B^r;
   (b) *All honest users receive the block* B^{r-1} *before waiting time* Λ *ends in round* r *step* 1 *so that each bid* b *in* B^r *satisfies* $\mathsf{bVrfy}(\mathsf{b}, C) = 1$ *and that any* b' *in the chain* $\mathcal{C}^r$ *with* $\mathsf{b}' \neq \mathsf{b}$ *corresponds to a serial number different from that of* b.
   (c) *Let* z *denote the number of bids issued by honest users that are not in the chain.If* $z \geq L$ *and* B^r *is nonempty, then* B^r *contains at least* $\mathfrak{h}z$ *bids.*

(2) *If r-leader* ℓ *exists, then the following holds.*
   – *The leader is* ℓ *and generates* $B^r = B^r_\ell$. *All honest users are sure about* B^r *in the time interval* I^{r+1} *and* $T^{r+1} \leq T^r + 5\lambda + 2\Lambda$;
   – *The block* B^r *is nonempty. If* ℓ *is honest, then the set of bids in* B^r *consists of all bids in* ℓ*'s mempool* MP^r_ℓ.

(3) *If r-leader does not exist, then all honest users are sure about* B^r *in the time interval* I^{r+1}. *Moreover, one of the following two cases happens.*
   – *If every verifier* $i \in HSV^{r,4}$ *has* $g_i \leq 1$, *then the Aucrand round* r *finishes in step* 6 *with an empty block and* $T^{r+1} \leq T^r + 8\lambda + 2\Lambda$.

– *Otherwise, i.e., some verifier* $i \in HSV^{r,4}$ *has* $g_i = 2$*, then Aucrand round* r *finishes before step* $6 + 3 \cdot L^r$ *and* $T^{r+1} \leq T^r + (6L^r + 8)\lambda + 2\Lambda$*, given that* $6 + 3 \cdot L^r$ *is smaller than the maximal number of steps allowed by the protocol designer. Here* L^r *denotes the random variable representing the number of Bernoulli's trials needed to see a* 1.

(4) *Assume the good case happens in rounds* $r-1$ *and* r*. The probability of some honest user becoming* r*-leader is* $\geq h^2(1 + h - h^2)$.

In the proof of this theorem, we use induction on rounds. The induction hypothesis is that all honest users are sure about the same $B^{r'-1}$ in the time interval $I^{r'}$ for $r' \in [r]_0$. We postpone the proof to Appendix E.2.

We are to show (2) and (a) under the assumption of (2). For this, we first propose an analogue of [10, Lemma 5.6]. (3) and (a) in this case can be proved following the same method as the one in the proof of Algorand's soundness lemma [10, Section 5.5]. This is because: (i) Algorand's soundness lemma relies on Algorand protocol for steps $s \geq 3$; (ii) the difference between the sidechain protocol and Algorand only comes from steps 1 and 2. By the hypothesis of (4), (4) can be proved following the same method as the method in the proof of [10, Lemma 5.5]. We omit the proof of (3) and (4). In the end, we show (b) and (c).

Lemma 16 (cf. [10, Lemma 5.6])**.** *Assume that all honest users are sure about the same* $B^{r'-1}$ *in the time interval* $I^{r'}$ *for* $r' \in [r]_0$*. Then for round* r *step* $s \geq 1$*, the following properties hold.*

(1) *For any honest user* i*, it holds that* $\alpha_i^{r,1} \in I^r$*,* $\alpha_i^{r,2} = \beta_i^{r,1} = \alpha_i^{r,1} + \Lambda$*,* $\alpha_i^{r,3} = \beta_i^{r,2} = \alpha_i^{r,1} + \lambda + 2\Lambda$ *and for* $s \geq 2$*,* $\beta_i^{r,s} \leq \alpha_i^{r,1} + (2s-3)\lambda + 2\Lambda$.

(2) *For any two honest users* i *and* i'*, it holds that* $|\beta_i^{r,s} - \beta_{i'}^{r,s}| \leq \lambda$.

(3) *If an honest user* i *has waited for the maximal amount of time required by Aucrand step* $s \geq 2$*, then before finishing her step* s*,* i *has received all messages sent by all honest verifiers in* $HSV^{r,s-1}$.

Proof. By the definition of Aucrand, the equalities and the inequality holds. By the hypothesis, we have $\alpha_i^{r,1} \in I^r$. This shows (1).

We show (2) and (3) by induction. The base case consists of steps $1, 2$. If i is an honest user, then $\beta_i^{r,1} = \alpha_i^{r,1} + \Lambda$. Hence, (2) for step 1 straightforwardly follows from $\alpha_i^{r,1} \in I^r$ for any honest user i. As for (2) for step 2, we note that the verifiers in $HSV^{r,2}$ has to wait $\lambda + \Lambda$ so that she can select the leader. Hence, $\beta_i^{r,2} = \alpha_i^{r,2} + \lambda + \Lambda$ for any $i \in HSV^{r.2}$ and (2) for step 2 follows from (1). We show (3) for step 2. For this, we show $\beta_i^{r,2} \geq \beta_{i'}^{r,1} + \Lambda$ for any honest user i'. By (1), we have $\beta_i^{r,2} = \beta_i^{r,1} + \lambda + \Lambda$. By (2) for step 1, we have $|\beta_i^{r,1} - \beta_{i'}^{r,1}| \leq \lambda$. Hence, $\beta_{i'}^{r,1} \leq \beta_i^{r,1} + \lambda$ and the desired inequality follows.

We show (2) and (3) for step $s \geq 3$ assuming (2) and (3) for step $s-1$. We show (2). Let i' be any honest user. As $\alpha_j^{r,s} = \beta_j^{r,s-1}$ for any honest user j, the induction hypothesis implies $|\alpha_i^{r,s} - \alpha_{i'}^{r,s}| \leq \lambda$. Hence,

$$\alpha_{i'}^{r,s} \leq \alpha_i^{r,s} + \lambda. \tag{E.1}$$

We may assume $\beta_i^{r,s} \le \beta_{i'}^{r,s}$. First, assume that i has waited for the maximal amount of time required by step s. Hence,

$$\begin{aligned}\beta_{i'}^{r,s} &\le \alpha_{i'}^{r,s} + 2\lambda && \text{(by Aucrand step } s\text{)}\\ &\le \alpha_i^{r,s} + \lambda + 2\lambda && \text{(by Eq. E.1)}\\ &= \beta_i^{r,s} + \lambda,\end{aligned}$$

as desired.

It suffices to show the case where i ends step s before waiting for the maximal amount of time. If $s = 3$, by Aucrand step 3, this happens when i has received $\ge t_H$ votes on $v = (H(B_\ell^r), \ell)$ for some user ℓ. As i has helped propagate the received messages corresponding to the $\ge t_H$ votes. The user i' receives all these messages before $\beta_i^{r,3} + \lambda$ and hence finishes her step 3 before $\beta_i^{r,3} + \lambda$. The desired inequality hence follows.

If $s = 4$, by Aucrand step 4, one of the following cases happens

$$\begin{cases} g_i = 2 \quad i \text{ received } \ge t_H \text{ votes on } v = (H(B_\ell^r), \ell) \text{ for some user } \ell; \\ g_i = 0 \quad i \text{ received } \ge t_H \text{ votes on } v = \perp. \end{cases} \tag{E.2}$$

In both cases, i has helped propagate the messages corresponding to the $\ge t_H$ votes. The user i' receives these messages before $\beta_i^{r,4} + \lambda$ and hence finishes her step 4 before $\beta_i^{r,4} + \lambda$. The desired inequality follows.

If $s \ge 5$ and i has received $\ge t_H$ votes for the same value v, then the value v might be

$$v = \begin{cases} (0, (H(B_\ell), \ell)) & \text{in step } 5 \mod 3, \text{ for some user } \ell; \\ 1 & \text{in step } 5,\ 6,\ \text{or } 7 \mod 3; \\ 0 & \text{in step } 6 \text{ or } 7 \mod 3. \end{cases}$$

In all cases, i helped propagate the messages corresponding to the $\ge t_H$ votes. The user i' receives these messages before $\beta_i^{r,s} + \lambda$ and hence finishes her step s before $\beta_i^{r,s} + \lambda$. The desired inequality follows.

Finally, we show (3) for step s. Now $\beta_i^{r,s} = \alpha_i^{r,s} + 2\lambda$. For any verifier $i' \in HSV^{r,s-1}$, we have $\beta_{i'}^{r,s-1} = \alpha_{i'}^{r,s}$, By Eq. E.1, we have

$$\beta_{i'}^{r,s-1} + \lambda = \alpha_{i'}^{r,s} + \lambda \le \alpha_i^{r,s} + 2\lambda = \beta_i^{r,s},$$

i.e., before the finish of i's step s, i receives all messages from i' if i waits for the maximal amount of time 2λ required by Aucrand in step s, as desired. □

Lemma 17 (Completeness Lemma, cf. [10, Lemma 5.3])**.** *Assume all honest users are sure about the same $B^{r'-1}$ in the time interval $I^{r'}$ for $r' \in [r]_0$. If r-leader ℓ exists, then the properties* (a) *and* (2) *hold.*

Proof. As r-leader ℓ exists, the messages propagated by verifiers in $HSV^{r,2}$ contains $\ge t_H$ votes on $(H(B_\ell^r), \ell)$. By Lemma 16(1), for any $i \in HSV^{r,2}$, we have

$$\beta_i^{r,2} \le T^r + 2\lambda + 2\Lambda.$$

By Lemma 16 (3), before waiting the required maximal amount of time 2λ, each verifier in $HSV^{r,3}$ receives $\geq t_H$ votes on $(H(B^r_\ell), \ell)$ from verifiers in $HSV^{r,2}$. Hence, all verifiers in $HSV^{r,3}$ set $v = (H(B^r_\ell), \ell)$ and propagate messages embedding v. As the propagation of short messages costs time λ, we have for $i \in HSV^{r,3}$

$$\beta_i^{r,3} \leq T^r + 3\lambda + 2\Lambda.$$

Before waiting the required maximal amount of time (Lemma 16 (3)), each verifier in $HSV^{r,4}$ receives $|HSV^{r,3}|$ votes on the value $(H(B_\ell), \ell)$ from verifiers in $HSV^{r,3}$. As $t_H < |HSV^{r,3}|$, there are $\geq t_H$ votes on $(H(B^r_\ell), \ell)$. By Aucrand step 4, each verifier $i \in HSV^{r,4}$ sets $g_i = 2$ and $b_i = 0$. Then i propagates messages embedding $(0, (H(B^r_\ell), \ell))$. We have

$$\beta_i^{r,4} \leq T^r + 4\lambda + 2\Lambda.$$

Before waiting the required maximal amount of time, each verifier in $HSV^{r,5}$ receives $|HSV^{r,4}|$ votes on $(0, (H(B^r_\ell), \ell))$ from verifiers in $HSV^{r,4}$. As $t_H < |HSV^{r,4}|$, there are $\geq t_H$ votes on $(0, (H(B^r_\ell), \ell))$. By Aucrand step 5, each verifier $i \in HSV^{r,5}$ enters Ending Condition 0 and forms the set $CERT^r$ using messages embedding $(0, (H(B^r_\ell), \ell))$. The verifier i is sure about $B^r = B^r_\ell$ before $T^r + 5\lambda + 2\Lambda$ and propagates $CERT^r$. Hence,

$$T^{r+1} \leq T^r + 5\lambda + 2\Lambda$$

and all honest users end round r in the interval I^{r+1}. Hence, (b) and the first point in (2) follow. The second point in (2) follows immediately from the protocol. □

Lemma 18. *Assume that all honest users are sure about the same $B^{r'-1}$ in the time interval $I^{r'}$ for $r' \in [r]_0$. Then, the properties* (b) *and* (c) *hold.*

Proof. Admit that the property (a) holds. First assume that r-leader ℓ exists. Then, by the definition of r-leader, there is an verifier i in $HSV^{r,2}$ (in fact, $\geq t_H$ verifiers in $HSV^{r,2}$) who has received and checked ℓ's candidate block. By Lemma 3(2), i has received B^{r-1} before finishing her step s. For each bid b in B^r, i has checked b satisfying the property (b) following $\mathsf{VrfyMsg}$ (Algorithm 3), using the set of commitments C and the blocks $B^{r'}$ for $r' \in [r-1]_0$. Hence, (b) follows. By Lemma 16(3), i has received all candidate blocks from honest potential leaders. For any honest potential leaders i' and the honest verifier i, their pools of serial numbers $\mathsf{SP}^r_{i'}$ and SP^r_i contain all serial numbers corresponding to all bids proposed by honest users that are not in the chain. Hence, i sets $y \geq |\mathsf{SP}^r_{i'} \cap \mathsf{SP}^r_i| \geq z \geq L$ (y defined before Lemma 4). As ℓ's candidate block B^r_ℓ is qualified by i, this candidate block contains $\geq \mathfrak{h}z$ bids. Hence, (c) follows.

For the case where r-leader does not exist, if $g_i < 1$ for all verifiers $i \in HSV^{r,4}$, then the block is empty by Theorem 1(3) and there is nothing to prove. We show (b) and (c) for the case where $g_i = 2$ with $v = (H(B^r_\ell), \ell)$ for some $i \in HSV^{r,4}$. By the previous paragraph, it suffices to show that there exists a verifier in $HSV^{r,2}$ who propagated her message embedding v. As $g_i = 2$, in i's view, v has got

$\geq t_H$ votes from $SV^{r,3}$. As $t_H > |MSV^{r,3}|$, there exists a verifier $i' \in HSV^{r,3}$ who propagated messages embedding v. In the view of i', v has got $\geq t_H$ votes from $SV^{r,2}$. As $t_H > |MSV^{r,2}|$, there exists a verifier in $HSV^{r,2}$ who propagated her message embedding v, as desired. □

E.3 The r-Election

The r-election as an extensive game. We follow the notions in [35, Definition 89.1] to define the following extensive game with perfect information. Its strategic form may be regarded as the strategic game in Definition 6. In fact, there is a slight difference between the definition of r-candidates and r-villains. However, after admitting (A.1) and (A.2), we regard that the strategic form of the game below is the strategic game in Definition 6.

Definition 19 (r-Election)**.** We first define the set of players $SV^{r,1}$ and the set of actions A_i available to each player $i \in SV^{r,1}$. Then we define the set of histories *HIS*, the player function P, and the utility function u_i for each player i in $SV^{r,1}$. The tuple $\left(SV^{r,1}, HIS, P, (u_i)_{i \in SV^{r,1}}\right)$ is an extensive game with perfect information, named the *r-election*.

(1) *The set of players* $SV^{r,1}$ is the set of potential leaders of Aucrand round r who have collected the mempools.

(2) For each player $i \in SV^{r,1}$, recall that MP_i^r denotes i's mempool defined in Step 1 in Section 3.2. Then *the set A_i of actions available to player i* in this game is a function $f_i : PK \setminus \{i\} \to \mathcal{MP}_i^r$, where $\mathcal{MP}_i^r$ denotes the set of all subsets of MP_i^r. One may regard $f_i(j)$ as the candidate block that the player i sends to the user j.

(3) To define the rest notions, we partition the players set $SV^{r,1}$ into two subsets.
- A player i is an *r-candidate* if [25]

$$\text{for } \forall\ j, j' \in PK \setminus \{i\},\ \text{we have } f_i(j) = f_i(j') \in \mathcal{MP}_i^r. \tag{E.3}$$

  For an r-candidate i, we let f_i denote $f_i(j)$. Following Algorithm 4, each verifier i' in H^2 is able to select a leader according to $f_j(i')$ for all $j \in SV^{r,1}$. If an r-candidate is the leader whose candidate block gets $\geq t_H$ votes from verifiers in H^2, we call this r-candidate the *r-leader*.
- A player j is said to be an *r-villain* if j is not an r-candidate, *i.e.*, $\exists\ i, i' \in PK$ with $i \neq i'$ such that $f_j(i) \neq f_j(i')$.

(4) Let I and J denote respectively the set of r-candidates and r-villains in $SV^{r,1}$. *The set of histories HIS* consists of $\emptyset$ and 2-tuples (a, b), where
- $a \in \prod_{i \in I} A_i$, where $\prod_{i \in I} A_i$ denotes the direct product of sets A_i.
- $b = \emptyset$ or $b \in \prod_{j \in J} A_j$.

(5) *The player function* P is defined by $P(\emptyset) = I$ and $P(a, \emptyset) = J$, $a \in \prod_{i \in I} A_i$.

[25] Being the r-leader (defined below) guarantees that all bids issued by i can be included in the round r block. By the utility function, an r-candidate is better off honestly propagating her candidate block to all users to get $\geq t_H$ votes from $HSV^{r,2}$.

(6) Finally, we are to define *the utility functions of players*. Let (a, b) be an element in $\prod_{i\in I} A_i \times \prod_{j\in J} A_j$ which is a terminal history.
- If some r-candidate i' becomes the r-leader, then for each r-candidate i ($i = i'$ or not), we let n_i denote the number of bids issued by i that are included in the r-leader's candidate block. Then put $u_i = n_i$ and $u_j = -|f_{i'}|$ for the utility of i and an r-villain j respectively.
- Otherwise, put $u_i = 0$ for all $i \in I$ and $u_j = 0$ for all $j \in J$.

Next, as the extensive game has perfect information, we work out its strategic form [35, Definition 94.1] below. The proof of the lemma is straightforward.

Lemma 20 (cf. Definition 6)**.** (1) *The strategy set S_i of each r-candidate $i \in I$ is the set $A_i = \mathcal{MP}_i^r$.*

(2) *The strategy set S_j of each r-villain $j \in J$ consists of functions $s_j : \prod_{i\in I} S_i \to A_j$. In particular, the strategy set of the r-Villain consists of functions*

$$(s_j)_{j\in J} : \prod_{i\in I} S_i \to \prod_{j\in J} A_j.$$

(3) *The tuple $\left(SV^{r,1}, (S_i)_{i\in SV^{r,1}}, (u_i)_{i\in SV^{r,1}}\right)$ is the strategic form of the r-election.*

Aucrand step 1 replaced with the r-election. We propose explicit relations between the r-election and Aucrand below. Specifically, we hope to consider the r-election as step 1 of the Aucrand round r. We admit that

(A.1) The r-Villain is controlled by the adversary if not in the r-election.

(A.2) An r-candidate behaves as an honest user before waiting time Λ ends in round r step 1. She also behaves as an honest user after round r step 1 if not corrupted.

Corollary 21 (Cf. Theorem 1)**.** *With the verifiers in $SV^{r,s}$ for $s \geq 2$ defined in Section* 4.1*, one of the following cases happens with overwhelming probability for each round $r \geq 0$.*

(1) *If an r-candidate ℓ becomes an r-leader, then the following holds.*
- *The leader is ℓ and generates $B^r = B_\ell^r$. All honest users are sure about the same B^r in the time interval I^{r+1} and $T^{r+1} \leq T^r + 5\lambda + 2\Lambda$;*
- *The block B^r is nonempty. The set of bids in B_ℓ^r consists of all bids in the set f_ℓ propagated by i.*

(2) *Otherwise, all honest users are sure about the same B^r in the time interval I^{r+1}. Moreover, one of the following two cases happens.*
- *If every verifier $i \in H^4$ has $g_i \leq 1$, then the Aucrand round r finishes in step 6 with an empty block and $T^{r+1} \leq T^r + 8\lambda + 2\Lambda$.*
- *Otherwise, i.e., some verifier $i \in H^4$ has $g_i = 2$, then Aucrand round r finishes before step $6 + 3 \cdot L^r$ and $T^{r+1} \leq T^r + (6L^r + 8)\lambda + 2\Lambda$, given that $6 + 3 \cdot L^r$ is smaller than the maximal number of steps allowed by the protocol designer.*

Proof. (1) straightforwardly follows from Theorem 1(2) (See Lemma 17).

We show (2). Note that an r-villain j's behaviors are restricted to her strategies in S_j. As the malicious users are Byzantine, there are the same or fewer outcomes if malicious users are replaced with r-villains in round r step 1. We are to construct the operations of r-villains and the adversary leading to the two outcomes in (2). Here, we empower the adversary, in each step, to select honest verifiers, *e.g.*, who finish their previous steps early (then, the adversary let them wait for the maximal amount of time required by the protocol). This is compatible with our rushing adversary setting. An adversary without this capability only has fewer advantages in fulfilling the following attack operations.

We assume that some r-villain j has the smallest hash. Assume *w.l.o.g.*, that $\mathsf{SP}^r_j \supset \mathsf{SP}^r_k$ for $k \in SV^{r,1} \setminus \{j\}$. Then we list the operations of the r-villains and the adversary so that the second point in (2) can be achieved.

Step 1 The r-villain j chooses the strategy that
- j sends (the message embedding) $\mathsf{B}^r_j = \mathsf{MP}^r_j$ to a_2 verifiers in H^2 for $t_H - |M^2| \leq a_2 \leq t_H - 1$;[26]
- j sends $\mathsf{B}^r_j = \emptyset$ to the rest verifiers in H^2.

Step 2 a_2 verifiers in H^2 propagate (the message embedding) the value $v \coloneqq (H(B^r_j), j)$ and the rest verifiers in H^2 propagate values $\neq v$. For M^2,
- (the adversary let) all verifiers in M^2 send v to a_3 verifiers in H^3 for $t_H - |M^3| \leq a_3 \leq t_H - 1$;
- all verifiers in M^2 send $\perp$ to the rest verifiers in H^3.

Some verifiers in H^2 may finish their step 2 at $T^r + 2\lambda + 2\Lambda$.

Step 3 a_3 verifiers in H^3 propagate the value v and the rest verifiers in H^3 propagate $\perp$. For M^3,
- all verifiers in M^3 send $\perp$ to a_4 verifiers in H^4 with $t_H - |M^4| \leq a_4 \leq t_H - 1$;
- all verifiers in M^3 send v to the rest verifiers in H^4;

The malicious verifiers in M^2 may send their messages so that all verifiers in H^3 wait for $3\lambda + 2\Lambda$. Some verifiers in H^3 may finish their step 3 at $T^r + 4\lambda + 2\Lambda$.

Step 4 By Eq. 4.1, we know $\frac{|H^3|}{2} < t_H - |M^3| \leq a_3$. Hence, there are a_4 verifiers in H^4 setting $g = 1$ and propagating $(1, v)$, and the rest $|H^4| - a_4$ verifiers in H^4 set $g = 2$ and propagate $(0, v)$. For M^4,
- $< t_H - a_4$ verifiers in M^4 send $(1, v)$ to a_5 verifiers in H^5 with $t_H - |M^5| \leq a_5 \leq t_H - 1$;
- all verifiers in M^4 send $(1, v)$ to the rest $|H^5| - a_5$ verifiers in H^5.

The malicious verifiers in M^3 may send their messages so that all verifiers in H^4 wait for 2λ in step 4. Some verifiers in H^4 may finish their step 4 at $T^r + 6\lambda + 2\Lambda = T^r + \lambda + (4 \cdot 2 - 3)\lambda + 2\Lambda$.

Step 5 As $|H^4| - a_4 + |M^4| \leq |H^4| - t_H + 2|M^4| < t_H$, no verifier in $H^5 \cup M^5$ is able to enter Ending Condition 0. By Coin-Fix-To-0, a_5 verifiers in H^5 propagate $(0, v)$ and $|H^5| - a_5$ verifiers in H^5 propagate $(1, v)$. For M^5,
- $< t_H - a_5$ verifiers in M^5 send $(0, v)$ to a_6 verifiers in H^6 with $t_H - |M^6| \leq a_6 \leq t_H - 1$;

[26] The "a_2 verifiers in H^2" here should be "verifiers in H^2 who own a_2 votes". We use "a_2 verifiers in H^2" for simplicity.

– all verifiers in M^5 send $(0, v)$ to $|H^6| - a_6$ verifiers in H^6;

The malicious verifiers in M^4 send their messages so that all verifiers in H^5 wait for 2λ in step 5. Some verifiers in H^5 may finish their step 5 at $T^r + 8\lambda + 2\Lambda = T^r + \lambda + (5 \cdot 2 - 3)\lambda + 2\Lambda$.

Step 6 As $|H^5| - a_5 + |M^5| \leq t_H$, no verifier in H^6 or M^6 is able to enter Ending Condition 1. By Coin-Fix-To-1, a_6 verifiers in H^6 propagate $(1, v)$ and there are $|H^6| - a_6$ verifiers in H^6 propagate $(0, v)$. For M^6,

– all verifiers in M^6 send $(1, v)$ to a_7 verifiers in H^7 with $t_H - |M^7| \leq a_7 \leq t_H - 1$;

– $< t_H - a_6$ verifiers in M^6 send $(1, v)$ to $|H^7| - a_7$ verifiers in H^7;

The malicious verifiers in M^5 may send their messages so that all verifiers in H^6 wait for 2λ in step 6. Some verifiers in H^6 may finish their step 6 at $T^r + 10\lambda + 2\Lambda = T^r + \lambda + (6 \cdot 2 - 3)\lambda + 2\Lambda$.

Step 7 a_7 verifiers in H^7 propagate $(1, v)$ and the rest verifiers in H^7 flip coin and decide either to propagate $(0, v)$ all-together or $(1, v)$ all-together (if the leader for the coin flip is honest).

Assume that $(1, v)$ are propagated by $\geq t_H - a_7$ who flipped the coin. Then round r will finish in step 9. Some verifier in H^7 may wait 2λ before she flips the coin. Each verifier in H^8, H^9 takes at most λ to finish step s for $s = 8$ or 9. All verifiers in H^9 may finish their step 9 before $T^r + 14\lambda + 2\Lambda = T^r + 4\lambda + 10\lambda + 2\Lambda$.

Thereafter assume that $(1, v)$ are propagated by $< t_H - a_7$ verifiers who flipped the coin.

– all verifiers in M^7 send nothing to a_8 verifiers in H^8 with $t_H - |M^8| \leq a_8 \leq t_H - 1$;

– all verifiers in M^7 send $(1, v)$ to the rest $|H^8| - a_8$ verifiers in H^8.

The situation in step 7 and 8 is the same as that of step 4 and 5. Then the adversary can take the same actions as in step 5 and 6. Hence, one may assume that Coin-Genuinely-Flipped Step 10 is needed and so on.

We calculate an upper bound for T^{r+1}. Recall that L^r denotes the random variable which is the number of Bernoulli's trials needed to see a 1 (in our case, this precisely means $(1, v)$ are propagated by $\geq t_H - a_s$ who flipped the coin with $s \equiv 7 \mod 3$). Some verifier in H^s may wait for 2λ in steps s for $3 \leq s \leq 4 + 3L^r$. Each verifier in H^s takes at most λ to finish steps s for $s = 5 + 3L^r, 6 + 3L^r$. All verifiers in H^{6+3L^r} may finish their step $6 + 3L^r$, which is a Coin-Fix-To-1 step before $T^r + (6L^r + 8)\lambda + 2\Lambda = T^r + \lambda + 2\lambda + ((3L^r + 4) \cdot 2 - 3)\lambda + 2\Lambda$.

Finally, to achieve the first point of (2), the r-villains and adversary follow the above Step 1 and Step 2 as slowly as possible. Some verifier in H^3 may finish their step 3 at $T^r + 4\lambda + 2\Lambda$. Note that the value $(H(B_j^r), j)$ gets a_3 votes from H^2 with $a_3 > |H^3|/2 > t_H/2$. Then by letting M^3 send nothing, all verifiers in H^4 wait 2λ and finish their step 4 at or before $T^r + 6\lambda + 2\Lambda$ with $g = 1$. As all verifiers in H^4 agree on $b = 1$, Aucrand round r finishes in step 6. Also, each honest user takes $\leq \lambda$ time to finish step 5 or 6. Hence, $T^{r+1} \leq T^r + 8\lambda + 2\Lambda$. □

E.4 Proof of the Equilibrium of the r-Election Game.

We admit $\mathsf{SP}^r_j = \mathsf{SP}^r_H$ for any $j \in J$. Recall the strategy profile $(s^*_k) = (s^*_k)_{k \in SV^{r,1}} \in \prod_{k \in SV^{r,1}} S_k$ below.

- For an r-candidate $i \in I$, we have $s^*_i = \mathsf{MP}^r_i$.
- For an r-villain $j \in J$, the function s^*_j is constant on $\prod_{i \in I} S_i$ and is a function satisfying

$$s^*_j : PK \setminus \{j\} \to \mathcal{MP}^r_j,\ s^*_j(i) = \begin{cases} \mathsf{MP}^r_j & i \in H^2_+; \\ \emptyset & i \in H^2_-, \end{cases} \tag{E.4}$$

where $H^2_+ \subsetneq H^2$ consists of players $i \in H^2$ such that $|H^2| - t_H < |H^2_+| < t_H$ and $H^2_- \coloneqq H^2 \setminus H^2_+$.

Theorem 2. *Assume $\geq \mathfrak{h}$ of series numbers in SP^r_H belong to SP^r_i for any $i \in I \cup H^2$ and $\mathfrak{h} \cdot |\mathsf{SP}^r_H| \geq L$. Then the profile (s^*_k) is an equilibrium for the r-election. Moreover, this profile is an equilibrium even if we regard that the r-Villain controls all r-villains.*

Proof. We first analyze the condition. Let i' be a verifier in H^2. For any $k \in SV^{r,1}$, let $\mathsf{S}^*_k(i')$ denote the set of serial numbers corresponding to the bids in $s^*_k(i')$. To select the leader, i' sets (see Eq. 3.1)

$$y^* = \max\left\{ |\mathsf{S}^*_k(i') \cap \mathsf{SP}^r_{i'}| \,\middle|\, k \in SV^{r,1} \right\}, \tag{E.5}$$

$$W^*_{i'} = \left\{ k \,\middle|\, |\mathsf{S}^*_k(i') \cap \mathsf{SP}^r_{i'}| \geq \mathfrak{h} y \right\}. \tag{E.6}$$

For any $k \in SV^{r,1}$ with $\mathsf{S}^*_k(i') = \mathsf{MP}^r_k$, we have by the hypothesis (cf. Proposition 4(2))

$$|\mathsf{S}^*_k(i') \cap \mathsf{SP}^r_{i'}| = |\mathsf{SP}^r_k \cap \mathsf{SP}^r_{i'}| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_H| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_{i'}| \geq \mathfrak{h} y^*$$

and hence $k \in W^*_{i'}$. Hence, we have the following three claims:

(C.1) Given (s^*_k), $W^*_{i'} = SV^{r,1}$ if $i' \in H^2_+$ and $W^*_{i'} = I$ if $i' \in H^2_-$ (See Eq. E.4).

(C.2) If s^*_k is replaced with $s_k \in S_k$ given $s^*_{-k} = (s^*_l)_{l \in -k}$, we have $W^*_{i'} = W_{i'}$, $W^*_{i'} = W_{i'} \cup \{k\}$, or $W^*_{i'} = W_{i'} \setminus \{k\}$ for any $i' \in H^2$, where $W_{i'}$ denotes the set obtained in Eq. E.6 after the replacement.

(C.3) For any $i \in I$, given s^*_i, we have $i \in W_{i'}$ for any $i' \in H^2$ even if s^*_{-i} is replaced with $(s_k)_{k \in -i} \in \prod_{k \in -i} S_k$.

We divide the remainder of the proof into two parts. Firstly, we show that s^*_i is a best response to s^*_{-i} for any $i \in I$. Secondly, we show that the tuple $(s^*_j)_{j \in J}$, regarded as a strategy of the r-Villain, is a best response to $(s^*_i)_{i \in I}$. In particular, this implies that s^*_j is a best response to s^*_{-j} for any $j \in J$.

(1) We need to show that the utility u_i does not increase for any $s_i \in S_i$, $s_i \subsetneq s^*_i$ given s^*_{-i}. Recall that $u_i = n_i$ if r-leader exists, or $u_i = -\tau$ otherwise.

Here, n_i denotes the number of bids committed by i added to the candidate block of the r-leader.

Let S_i be the set of serial numbers corresponding to bids in s_i and y the number defined as in Eq. E.5 with S_i^* replaced with S_i. For any $i' \in H^2$, we have $\mathsf{S}_i \cap \mathsf{SP}_{i'}^r \subsetneq \mathsf{S}_i^* \cap \mathsf{SP}_{i'}^r$. If $|\mathsf{S}_i \cap \mathsf{SP}_{i'}^r| \geq \hbar y$, then by (C.2) we have $i \in W_{i'} = W_{i'}^*$. Hence, the probability of i being the leader remains the same. Moreover, in this case, the probability that the r-leader exists also remains the same. Overall, replacing s_i^* with s_i does not change i's utility.

If $|\mathsf{S}_i \cap \mathsf{SP}_{i'}^r| < \hbar y$, then by (C.2) we have $W_{i'} = W_{i'}^* \setminus \{i\}$ for any $i' \in H^2$ and i has no chance to be the leader in the view of i'. Hence, the probability of i being the leader decreases. Moreover, in this case, the probability that the r-leader exists decreases because the fraction of r-candidates in $W_{i'}$ is smaller than that of $W_{i'}^*$. Overall, replacing s_i^* with s_i leads to a decrease of i's utility.

(2) Fix $(s_i^*)_{i \in I}$. By (C.3), after replacing $(s_j^*)_{j \in J}$ with $(s_j)_{j \in J} \in \prod_{j \in J} S_j$, we have $I \subset W_{i'}$ for any $i' \in H^2$. If no $j \in J$ has the smallest hash, then replacing $(s_j^*)_{j \in J}$ with $(s_j)_{j \in J}$ will not change r-Villain's utility. Hence, we may assume that some $j \in J$ has the minimal hash among all players in $SV^{r,1}$. If the r-Villain follows the strategy $(s_j^*)_{j \in J}$, then by (C.1) there are $|H_+^2|$ votes regarding j as the leader and $|H^2| - |H_+^2|$ votes from H_-^2 regarding other players as the leader. As $|H^2| - t_H < |H_+^2| < t_H$, there is no r-leader (See Definition 19(3)) and $u_j = 0$, *i.e.*, j's utility attains the maximal. Therefore, $(s_j^*)_{j \in J}$ is the best response to $(s_i^*)_{i \in I}$. □

E.5 Partial results in certain cases

Removing the condition "$\geq \hbar$ of serial numbers in SP_H^r belong to SP_i^r for any $i \in I \cup H^2$" turns out to be difficult. We are to show some partial results.

Lemma 22. *Let i be an r-candidiate. For any strategy $s_{-i} = (s_k)_{k \in -i} \in \prod_{k \in -i} S_k$, the strategy $s_i^* = \mathsf{MP}_i^r$ is a best response to s_{-i} if there is $i_0 \in I \setminus \{i\}$ such that $\mathsf{S}_i^* \subseteq \mathsf{S}_{i_0}$, where S_i^* and S_{i_0} denote respectively the sets of serial numbers corresponding to bids in s_i^* and s_{i_0}.*

Proof. The goal is to show that the utility u_i for an r-candidate i does not increase for any $s_i \in S_i$ with $s_i \subsetneq s_i^* = \mathsf{MP}_i^r$ given $s_{-i} = (s_k)_{k \in -i}$. We first analyze the condition. Let i' be any verifier in H^2. For $k \in SV^{r,1}$, let $\mathsf{S}_k(i')$ denote the serial numbers corresponding to the bids in $s_k(i')$. To select the leader, i' sets

$$y = \max\left\{ |\mathsf{S}_k(i') \cap \mathsf{SP}_{i'}^r| \,\middle|\, k \in SV^{r,1} \right\},$$
$$W_{i'} = \left\{ k \,\middle|\, |\mathsf{S}_k(i') \cap \mathsf{SP}_{i'}^r| \geq \hbar y \right\}.$$

Put $y^* \coloneqq y$ and $W_{i'}^* \coloneqq W_{i'}$ if $s_i = s_i^*$. By the existence of i_0, we know $y^* = y$. Hence, either $W_{i'}^* = W_{i'}$ or $W_{i'}^* = W_{i'} \cup \{i\}$. This implies that replacing s_i^* with s_i will not lead to a modification of $W_{i'}^* \setminus \{i\}$ for $i' \in H^2$.

Similar to part (1) in the proof of Theorem 2, we show that for any $i' \in H^2$, the strategy s_i of i does not increase: (1) the probability of i being the leader in the view of i'; (2) the fraction of r-candidates in $W^*_{i'}$. In fact, by the existence of i_0, we have $y = y^* \geq |s^*_i| > |s_i|$ in the view of i'. Hence, $|s_i| \geq \mathfrak{h}y$ or $|s_i| < \mathfrak{h}y$ given $|s^*_i| \geq \mathfrak{h}y$. In both cases, (1) and (2) hold. □

We are to give an example where if an r-candidate i does not choose $s^*_i = \mathsf{MP}^r_i$, *i.e.*, not behave the same as an honest potential leader, then her strategy may not be a best response to s^*_{-i}. By the committee selection process of Aucrand, the adversary and the r-Villain may realize the strategy in Eq. E.4 as below (we admit that the adversary controls the r-Villain except in the r-election as in (A.1)):

1. Choose a subset HPK_+ of HPK such that $|HPK_+| = |HPK| \cdot a/|H^2|$, where HPK denotes the set of honest users in PK. Send all committed bids to HPK_+ and send no bid to $HPK_- \coloneqq HPK \setminus HPK_+$ before honest users collect SP^r.
2. Each r-villain $j \in J$ chooses the strategy

$$s^*_j(i) = \begin{cases} \mathsf{MP}^r_j & i \in HPK_+; \\ \emptyset & i \in HPK_-. \end{cases}$$

By the behavior of the adversary, we have $\mathsf{SP}^r_i = \mathsf{SP}^r_{i'}$ if i, i' both in HPK_+ or both in HPK_-. We admit $H^2_+ = HPK_+ \cap H^2$ and $H^2_- = HPK_- \cap H^2$. Put $I_+ \coloneqq HPK_+ \cap I$ and $I_- \coloneqq HPK_- \cap I$.

We may regard $|I_+| \geq 2$. Then by Lemma 22, for each $i \in I$, the strategy $s^*_i = \mathsf{MP}^*_i$ is a best response to $(s_k)_{k \in -i} \in \prod_{k \in -i} S_k$ if $s_k = \mathsf{MP}^r_k$ for $k \in I$. Next, we show that $s_i \subsetneq s^*_i$ may not be a best response.

Proposition 23. *Assume that the adversary behaves as above. For an r-candidate $i \in I_-$ and the strategy profile s^*_{-i} defined above, put $c = |\mathsf{SP}^r_i|/|\mathsf{SP}^r_j|$ for $j \in J$ and assume $\mathfrak{h} \leq c \leq 1$. Then the strategy $s_i \subsetneq \mathsf{MP}^r_i$ with $|s_i| = c' \cdot |\mathsf{SP}^r_i|$ for $c' < \mathfrak{h}/c$ is not a best response to s^*_{-i}.*

Proof. For a verifier $i' \in H^2$, retain the notations y, $W_{i'}$, y^*, and $W^*_{i'}$ defined in Lemma 22. If i chooses $s^*_i = \mathsf{MP}^r_i$, then we have $i \in W^*_{i'}$ for $i' \in H^2_+$ because $y^* = |\mathsf{SP}^r_j|$ and $|\mathsf{SP}^r_i \cap \mathsf{SP}^r_{i'}| = |\mathsf{SP}^r_i| = c \cdot |\mathsf{SP}^r_j| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_j|$. Also, we have $i \in W^*_{i'}$ for $i' \in H^2_-$ as $|\mathsf{SP}^r_i \cap \mathsf{SP}^r_{i'}| = |\mathsf{SP}^r_{i'}|$. Consequently, $i \in W^*_{i'}$ for any $i' \in H^2$. The r-candidate i is the leader in the view of i' if she has the minimal hash.

Each verifier $i' \in H^2_+$ sets $y = |\mathsf{SP}^r_j|$ as $s_j(i') = \mathsf{MP}^r_j$ and $\mathsf{SP}^r_{i'} = \mathsf{SP}^r_j$ for $j \in J$. Assume that i chooses the strategy s_i. As

$$|s_i| = c' \cdot |\mathsf{SP}^r_i| < \frac{\mathfrak{h}}{c} \cdot |\mathsf{SP}^r_i| = \mathfrak{h} \cdot |\mathsf{SP}^r_j| = \mathfrak{h}y,$$

we have $i \notin W_{i'}$. Moreover, for any $i'' \in I \setminus \{i\}$ and $i' \in H^2_+$, we have $i'' \in W_{i'}$ and $i'' \in W^*_{i'}$ because $y = y^* = |\mathsf{SP}^r_j|$ and $|s^*_{i''}| = c \cdot |\mathsf{SP}^r_j| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_j| = \mathfrak{h}y$. Hence, $W_{i'} = W^*_{i'} \setminus \{i\}$ for $i' \in H^2_+$. Notice $\mathsf{SP}^r_{i''} = \mathsf{SP}^r_{i'}$ for $i' \in H^2_-$. We then have $i'' \in W_{i'}$ and $i'' \in W^*_{i'}$. Hence, $W_{i'} = W^*_{i'}$ or $W_{i'} = W^*_{i'} \setminus \{i\}$ for $i' \in H^2_-$.

As $|H^2| - t_H < |H^2_+| < t_H$, i can never be the r-leader, and the probability that r-leader exists decreases. □

Next, we work out a counter-example.

Proposition 24. *Assume that the adversary behaves as above. The strategy profile (s^*_k) is an equilibrium for the r-election. If we regard that the r-Villain controls all r-villains, this profile is an equilibrium but is* not *a subgame equilibrium.*

Proof. Throughout the proof, we regard the r-Villain controls all r-villains. We divide the proof into two parts. Firstly, we show that the profile (s^*_k) is an equilibrium.[27] Secondly, we show that (s^*_k) is not a subgame equilibrium.

(1) By Lemma 22, for an r-candidate i, we have that s^*_i is a best response to s^*_{-i}.

For a verifier $i' \in H^2$, retain the notations y and $W_{i'}$ defined in Lemma 22. Fix $(s^*_i)_{i\in I}$. We show that replacing $(s^*_j)_{j\in J}$ with $(s_j)_{j\in J} \in \prod_{j\in J} S_j$ does not increase the utility of an r-villain which equals the utility of the r-Villain. If $|\mathsf{SP}^r_i| \geq \mathfrak{h} \cdot |\mathsf{SP}^r_j|$ for $j \in J$ and some $i \in I_-$ (Hence, for any $i \in I$, by assumption), we have $i \in W_{i'}$ for any $i \in I$ and any $i' \in H^2$. If no r-villain in J has the smallest hash value, then whatever strategies the r-Villain chooses will not change her utility. Hence, we may assume that some $j \in J$ has the minimal hash value among all players in $SV^{r,1}$. If each $j \in J$ chooses s^*_j in Eq. E.4, then verifiers in H^2_+ regard j as the leader and verifiers in H^2_- regard other players as the leader. As $|H^2| - t_H < |H^2_+| < t_H$, there is no r-leader (See Definition 19(3)). Hence, $u_j = 0$, *i.e.*, j's utility attains the maximal for all $j \in J$.

Otherwise, *i.e.*, $|\mathsf{SP}^r_i| < \mathfrak{h} \cdot |\mathsf{SP}^r_j|$ for all $i \in HPK_-$, we have $i \notin W_{i'}$ for $i \in I_-$ and $i' \in H^2_+$. If some $i \in I_-$ has the smallest hash value and some r-villain $j \in J$ has the second smallest hash value, then following Eq. E.4 leads to $u_j = 0$ and the utility of the r-Villain attains the maximal. For the case where an r-candidate i'' owns the second smallest hash, if $i'' \in I_+$, then $i'' \in W_{i'}$ for any $i' \in H^2$ and whatever strategies the r-Villain chooses will not change her utility. On the other hand, if $i'' \in I_-$ so that $|\mathsf{SP}^r_{i''}| < \mathfrak{h} \cdot |\mathsf{SP}^r_j|$, then we essentially go back to the situation above. Finally, it suffices to consider the case where none of $i \in I_-$ has the smallest hash. For any $i \in I_+$, $i' \in H^2$, and $j \in J$, because $\mathsf{SP}^r_i = \mathsf{SP}^r_j \supseteq \mathsf{SP}^r_{i'}$, we know $i, j \in W_{i'}$. Hence, this case is similar to the one in the previous paragraph.

(2) Admit $\mathfrak{h} = 2/3$. We construct tuples $(s_i)_{i\in I} \in \prod_{i\in I} S_i$ and $(s_j)_{j\in J} \in \prod_{j\in J} S_j$ so that given $(s_i)_{i\in I}$, replacing $(s^*_j)_{j\in J}$ with $(s_j)_{j\in J}$ increases the probability of some $j \in J$ being the leader. This means that $(s^*_j)_{j\in J}$ is not a best response to $(s_i)_{i\in I} \in \prod_{i\in I} S_i$. Assume $2/3 \cdot |\mathsf{SP}^r_j| = |\mathsf{SP}^r_i|$ for $i \in HPK_-$.[28] Let P denote a subset of SP^r_i such that $|P| = 1/2 \cdot |\mathsf{SP}^r_j \cap \mathsf{SP}^r_i|$ for $i \in HPK_-$. Put $Q \coloneqq P \cup (\mathsf{SP}^r_j \setminus \mathsf{SP}^r_i)$. Consider $(s_i)_{i\in I}$ where s_i is the subset of MP^r_i such that

[27] This implies that (s^*_k) is an equilibrium for the r-election if we do not regard that all r-villains controlled by the r-Villain.

[28] This implies that the assumption in Theorem 2 holds.

the set of serial numbers corresponding to bids in s_i equals Q (resp. P) for all $i \in I_+$ (resp. $i \in I_-$).

Thereafter, assume that r-candidates choose the strategy $(s_i)_{i\in I}$. If the r-Villain follows $(s_j^*)_{j\in J}$, we have $i \in W_{i'}$ for $i \in I_+$ and $i' \in H_+^2$ because of $|Q \cap \mathsf{SP}_{i'}^r| = 2/3 \cdot |\mathsf{SP}_{i'}^r|$ and $\mathsf{SP}_{i'}^r = \mathsf{SP}_j^r$ for $j \in J$. We also have $i \in W_{i'}$ for $i' \in H_-^2$ because of $Q \cap \mathsf{SP}_{i'}^r = P \cap \mathsf{SP}_{i'}^r$ and $s_j^*(i') = \emptyset$ for $j \in J$. If some $i \in I_+$ has the smallest hash value among all players in $SV^{r,1}$, then i is the r-leader.

On the other hand, if each $j \in J$ follows a strategy s_j satisfying

$$s_j : PK \setminus \{j\} \to \mathcal{MP}_j^r,\ s_j(i) = \begin{cases} \emptyset & i \in HPK_+; \\ \mathsf{MP}_j^r & i \in HPK_-, \end{cases}$$

then each $i' \in H_-^2$ sets $y = |\mathsf{SP}_j^r \cap \mathsf{SP}_{i'}^r|$ as $H_-^2 = HPK_- \cap H^2$. Notice $|Q \cap \mathsf{SP}_{i'}^r| = |P \cap \mathsf{SP}_{i'}^r| = 1/2 \cdot |\mathsf{SP}_j^r \cap \mathsf{SP}_{i'}^r|$ for $i' \in H_-^2$. We have $i \notin W_{i'}$ for $i \in I$ and $i' \in H_-^2$. As $|H^2| - t_H < |H^2| - |H_-^2| < t_H$, there is no r-leader and $u_j = 0$ for any $j \in J$ with probability 1. □