A Smart Contract Oracle for Approximating Real-World, Real Number Values

William George
Kleros Cooperative, Montreal, Canada
william@kleros.io

Clément Lesaege
Kleros Cooperative, Lisbon, Portugal
clement@kleros.io

Abstract

A key challenge of smart contract systems is the fact that many useful contracts require access to information that does not natively live on the blockchain. While miners can verify the value of a hash or the validity of a digital signature, they cannot determine who won an election, whether there is a flood in Paris, or even what is the price of ether in US dollars, even though this information might be necessary to execute prediction market, insurance, or financial contracts respectively.

A number of promising projects and research developments have provided a better understanding of how one might construct a decentralized, binary oracle - namely an oracle that can respond by one of two possibilities, typically "yes" or "no", even while not requiring the interaction of a trusted third party. In this work, we extend these ideas to construct a general-purpose, decentralized oracle that can estimate the value of a real-world quantity that is in a dense totally ordered set, such as $\mathbb{R}$. In particular, this proposal can be used to estimate real number valued quantities, such as required for a price oracle. We will establish a number of desirable properties about this proposal. Particularly, we will see that the precision of the output is tunable to users' needs.

2012 ACM Subject Classification Theory of computation → Algorithmic game theory and mechanism design; Security and privacy → Distributed systems security

Keywords and phrases price oracle, Ethereum, blockchain

Digital Object Identifier 10.4230/OASIcs.Tokenomics.2019.6

1 Introduction and related work

Blockchains and, specifically, smart contract platforms such as Ethereum [2], provide significant opportunities for systems that transfer value in a trustless way. However, the inability of blockchains to natively observe events in the outside world has limited this potential. While miners can verify computations, such as required for validating a digital signature, as part of the protocol they perform, this protocol generally does not have access to off-chain information, such as weather data or even prices of blockchain-based assets in USD terms that may be required for contracts such as flood insurance or financial contracts to perform properly. Indeed, while new models of economic relationships have been seen to be facilitated by smart contracts [20, 11], such smart contracts will likely often require access to off-chain information. Already in the Ethereum white paper [2], the need for oracles, i.e. mechanisms that can import external information on-chain, is discussed as necessary to overcome the limitation of "value-blindness" of financial contracts towards crypto-assets. The limitations on the types of applications that are possible on smart contract platforms that are imposed by the difficulty in obtaining adequate oracles have since remained an active source of discussion in the blockchain community [9, 13].

Town Crier [21] and TLS Notary [6] allow for demonstrating on-chain that some information has been published on a given TLS enabled website. However, this approach requires that the website be trusted to honestly provide the required information; hence it is not appropriate for all use cases. Furthermore, this style of oracle can be vulnerable to insider attacks where a malicious employee of whatever entity controls the website can influence the information that is used on-chain. ChainLink [14], which has acquired Town Crier [12], proposes a model in which data drawn from multiple third party sources can be aggregated and a reputation system is used to evaluate different sources.

A contrasting approach is taken by decentralized oracles, namely oracles that do not depend on any trusted third party with special privileges. These systems typically involve setting up some economic game. Then, the incentive structure of this game is designed so that if participants follow their incentives, the oracle will produce correct answers.

The most successful example of a price oracle to date is likely that which is used by Maker DAO [16] so that its stablecoin Dai can remain pegged to the US dollar. As Maker DAO is based on a system of collateralization, and collateral is in ether, an oracle that can give the price of ether in USD is required. Maker DAO uses a median of values provided by trusted authorities such as leading exchanges, which are chosen (and can be replaced) by MKR token holders. Thus, this oracle is decentralized in the sense that it is ultimately responsive to token holders, albeit via a delegated system. It is worthwhile to note that, in this case, MKR holders have a strong incentive to choose good oracles as the usefulness of Dai as a stablecoin drives demand for MKR tokens, which must be burned to pay a fee when recovering collateral. While this system has contributed to the impressive stability of Dai, even in the recent cryptocurrency bear market, it is not clear how to generalize this idea to price oracles not built around a widely-used stablecoin.

Of particular note are the existing oracles based on the concept of Schelling points [18]. Here, users submit answers for what the output of the oracle should be and are rewarded if they are coherent with the majority and/or punished if they are incoherent. An early proposal for a Schelling point based oracle was Truthcoin [19]. The underlying idea, is then that users will submit true answers because they expect other users to also submit true answers. An attempt to apply this idea to price oracles discussed in [10] is to have users submit a value and then outputting the median submission. Then submitters are penalized if their submission falls outside of the 25th to 75th percentile range of all submissions, ideally encouraging coherence around the true result. As already discussed in [10], this first version of a Schelling point based price oracle is potentially vulnerable to "micro-cheating" as submitters risk little penalty if they provide small variations on the true value, as long as this variation is not too extreme. Such "micro-cheating" may allow an attack to nudge the output of the oracle and potentially affect the decision making of other submitters, leading to more substantially deviations over time.

A notable example of a system that builds upon the ideas of Truthcoin is Augur [17]. Augur provides oracles for prediction markets by allowing holders of the Augur token REP to dispute responses provided by default responders designated by the creator of the market. Augur allows question askers to create markets where the answer can be binary, multiple choice, or scalar (i.e. drawn from $\mathbb{R}$). Notably, when creating a scalar prediction market, the asker must currently specify a minimum and maximum value, as well as a precision, which somewhat limits the flexibility of such markets. Moreover, note that Augur itself requires knowledge of the price of the Augur token REP in order to determine if its fees should be adjusted up or down in order to perform a "market cap nudge" so that buying a majority of the REP to perform a 51% attack is financially not worthwhile relative to the amount of

value at stake in Augur prediction markets. This would be a natural use case for Augur's own scalar oracle. However, for the moment, Augur has opted to use a delegated system based on a collection of trusted third parties to determine the price of REP for these fee adjustments and considers eventually using its scalar oracle for this purpose an "active area of research" [5, 17].

Similar to Augur, Gnosis [4] offers a prediction market system based on oracles. Gnosis is "oracle agnostic," meaning that a Gnosis prediction market contract can reference any oracle [8]. That said, the Gnosis team has developed oracles themselves which can be used for their markets, such as their "ultimate oracle" mechanism [3]. Gnosis allows for prediction markets based on scalar values such as prices, by proposing ultimately binary questions such as whether the price will rise or fall [8].

Kleros [15] is a dispute resolution system on Ethereum based on the idea that parties can enter into business relationships for which any financial transfers are held in escrow in an Ethereum contract. Then, in case of dispute between the parties, a number of "jurors" are randomly drawn via a token weighted system to rule on the case and are incentivized via a Schelling point based mechanism. Already, such a system is a kind of oracle as it brings knowledge of the honest party in these cases on-chain, and the Kleros white paper [15] envisages asking jurors a wide variety of questions, so that Kleros can act as a general oracle. At its current stage of development [1], Kleros allows its jurors to rule between a finite set of predetermined options, particularly allowing binary choices. A notable feature of Kleros, as the cases that are considered by Kleros may require a substantial per-juror effort to be analyzed, is that it includes an appeal system so that juror effort is minimized even as the security against potential attacks should scale with the number of jurors that would be involved in a potential appeal.

Finally, ASTRAEA [7] proposes an oracle capable of providing a binary outputs and provides a rigorous analysis of the security properties of their system. ASTRAEA makes use of two groups – "voters" and "certifiers" – with different incentive structures that must agree for the oracle to return a value. This departs somewhat from the structure of the Schelling point based systems described above, but uses similar ideas.

2 Our contribution

We assume the existence of a decentralized, general purpose oracle that is capable of deciding between binary propositions, namely it is capable of producing true answers to statements about the external world such that the response is either "yes" or "no." For example, one could use Kleros [15], Augur [17], or ASTRAEA [7] to play this role, inheriting their respective security models and guarantees. Then we propose a way to extend such a binary oracle into an oracle capable of producing an element in a dense totally ordered set - namely a totally ordered set such that if x and y are in the set, there exists some z in the set such that $x < z < y$. In particular, as the set of real numbers $\mathbb{R}$ is an example of such a set, our oracle can be used to determine the price of some asset in a way that can be used on-chain. This oracle remains decentralized and general purpose. We establish a number of desirable properties about this oracle, particularly that the ultimate precision of the output dynamically adjusts to the greatest level of precision that is demanded by a user that is ultimately determined to be honest. Hence, honest users can force very precise outcomes (see Theorem 11) while limiting the ability of hostile users to delay the functioning of the oracle or otherwise consume network resources (see Propositions 8 and 9).

3 Proposal

We take $\mathcal{S}$ to be a dense totally ordered set (such as $\mathbb{R}$).

3.1 Assumptions on the underlying binary oracle

We suppose we have access to a pre-existing oracle that can decide binary propositions about the real-world. Namely, suppose we have two statements $\mathcal{A}$ and $\mathcal{B}$, one of which corresponds to reality. Then the binary oracle:

$$\mathcal{O}_B : \{\text{possible submissions to a (cryptoeconomic) game}\} \to \{\mathcal{A}, \mathcal{B}\}$$

will return one of $\mathcal{A}$ or $\mathcal{B}$ in a way that can be computed on-chain, after a delay of t Ethereum blocks for the (cryptoeconomic) game to be played, and in exchange for paying some fee A.

Furthermore, we allow for the possibility that the binary oracle has an appeal mechanism, which may cause an additional time delay and additional fees. Namely, we suppose that any actor can appeal a ruling of the binary oracle in exchange for paying an additional fee, $f_{\mathcal{A},i}$ or $f_{\mathcal{B},i}$, when one is in the ith appeal round and is staking on the claim that $\mathcal{A}$ or $\mathcal{B}$ is true respectively. One might want to have $f_{\mathcal{A},i} \neq f_{\mathcal{B},i}$, for example, to require higher appeal fees for the side that lost the previous round. If this fee is paid on behalf of one "side," i.e. is staked on the truth of $\mathcal{A}$ or $\mathcal{B}$, the corresponding fee must be paid for the other side as well (with the potential for multiple actors to pay this fee collectively). If one side pays an appeal fee and not the other, the side that pays its fees is considered to be the result of the binary oracle. If both sides pay their fees, the binary oracle rules again (with the idea that more resources can be put towards this ruling, so ideally it is more likely to be correct). Whatever fees that are paid on behalf of the side that is the ultimate output of the oracle after all appeals are refunded, whereas fees paid on behalf of the other side are lost (with the potential that they are at least partially redistributed to the fee payers of the winning side).

Appeals result in delaying the result of the oracle by an additional t Ethereum blocks per appeal, up to some maximum number of appeal rounds. In Section 5 we will assume bounds on the growth of appeal fees and consider resulting bounds on the attacker's ability to delay the result of the oracle in terms of her resources.

This proposal was originally developed as an extension to Kleros [15]. As a result, our assumptions above on the structure of eventual appeals are modeled on the Kleros system. The Kleros fee model [1] is designed so that there is an incentive to pay fees on behalf of outcomes that one thinks are likely to ultimately to be chosen by the oracle because one can win some of the fees staked by the losing side. Indeed, Kleros envisages the participation of fee insurers whose economic model is to pay fees on behalf of cases they deem worthy, reducing the practical inconvenience of requiring both sides of the case to pay fees.

However, the successive dispute rounds used in Augur [17] can also be thought of as appeals satisfying this structure (taking two consecutive dispute rounds together and thinking of them as a single appeal), with their "forking market" representing a decision that has reached the maximum number of appeals. Moreover, our results should be adaptable to binary oracles with differing appeal systems. Indeed, one can recover the situation of a binary oracle without an appeal mechanism by just considering that the maximum number of appeals is zero. Hence our results apply equally to such systems.

3.2 Discussion of actors and attack model

The principal actors of our proposal, beyond whatever actors participate in the underlying binary oracle that is used, are respondents, who submit information about $v \in \mathcal{S}$, the value that the oracle is attempting to determine. Specifically each respondent submits an interval in

which they believe that v belongs. Respondents pay a deposit D which they risk losing if the information provided is ultimately judged to be incorrect, see Section 4. While respondents may obtain a reward if the information they provide is judged to be correct, we will see that they are not on average compensated by our system. Hence, we expect that the primary motivation of respondents is some external interest in the result the oracle produces. We will see, under some idealized assumptions about the performance of the $\mathcal{O}_B$, that it is sufficient for there to be a single honest respondent for our oracle to produce honest results, see Theorem 11. For an oracle that is part of a decentralized application with wide use, participation of respondents in these conditions is not an unrealistic assumption. However, an investigation of additional ways to incentivize the participation of respondents may be a subject for future work.

In this work, we will consider attacks from attackers that have the capacities of respondents. Namely, we will analyze attacks that submit malicious responses or call appeals in a hostile manner. Of course, the quality of the results of our real-valued oracle depends on the quality of the results of the underlying binary oracle. As we allow for the possibility of using any binary oracle, we do not directly consider an attack model where it is possible to corrupt the results of the binary oracle. However, in our results, it will be clearly indicated when one must make hypotheses about the accuracy of $\mathcal{O}_B$. Furthermore, we do not consider attacks on the underlying infrastructure of the smart contract platform, such as 51% attacks or network denial-of-service attacks on Ethereum.

3.3 Proposed oracle algorithm

The procedure we propose to approximate the true value of some quantity is based on a sort of modified binary search of the responses, where, rather than split the list of responses at the median when performing a comparison, we detect incoherences that prevent a consensus answer from being accepted and then take a comparison with respect to the median of the list of these incoherences. We are performing these operations on elements in $\mathcal{S}$, which a priori is not closed under averaging, so the normal median may not be defined. However, if we need to take the median of a set D with an even number of elements, namely in the case that requires computing an average, as $\mathcal{S}$ is a dense totally ordered set, we can find some (not necessarily unique) element z of $\mathcal{S}$ such that half of the elements of D are on either side of z. We suppose that for a given $\mathcal{S}$ one has some way of efficiently choosing such a z, and we consider it to be a median of D. In the remainder of this paper, in the context of generic dense totally ordered sets, we use the words median and average in this sense. In the case $\mathcal{S} = \mathbb{R}$, we take the normal median.

In detail, we consider the following:

▶ **Algorithm 1.** *Input: Each respondent $\mathcal{USR}_i$ submits two distinct values - a lower bound $l_i \in \mathcal{S}$ and an upper bound $u_i \in \mathcal{S}$, $l_i < u_i$, giving an interval (l_i, u_i) in which this respondent believes the true value of the question is located.*

- *Sort the lower bound responses into a list $\mathcal{L}$ and the upper bound responses into a list $\mathcal{U}$, where in each case identical values are considered as single elements.*
- *Compute the lists*

$$\mathcal{L}_0 = \{l_i \in \mathcal{L} : \exists u_j \in \mathcal{U},\ u_j \leq l_i,\ \nexists\ l_k \in [u_j, l_i) \cap \mathcal{L}\}$$

and

$$\mathcal{U}_0 = \{u_i \in \mathcal{U} : \exists l_j \in \mathcal{L},\ l_j \geq u_i,\ \nexists\ u_k \in (u_i, l_j] \cap \mathcal{U}\}\,.$$

- *Compute*

  $$\mathcal{C}_0 = \{median(l_i, u_j) : l_i \in \mathcal{L}_0,\ u_j \in \mathcal{U}_0,\ u_j \leq l_i,\ \nexists\ l_k \in [u_j, l_i) \cap \mathcal{L},\ \nexists\ u_k \in (u_j, l_i] \cap \mathcal{U}\}$$

  (So, if we considered $\mathcal{L}$ and $\mathcal{U}$ in the same line, essentially $\mathcal{L}_0$ would consist of lower bounds which have an upper bound to their immediate left and $\mathcal{U}_0$ would consist of upper bounds that have a lower bound to their immediate right. Then $\mathcal{C}_0$ consists of the midpoints between each of these pairs.)
- *If $\mathcal{C}_0 \neq \emptyset$*
  - *For each $z \in \mathcal{C}_0$ perform the following in parallel:*
    - *Ask the binary oracle $\mathcal{O}_B$ if*

      desired value $\leq z$

      or

      desired value $> z$.

    - *Allow appeals of their decision as necessary following the fee structure described in Section 3.1, where here the two sides are as follows:*

      desired value $\leq z$

      or

      desired value $> z$

      - *If one side pays its required fees but not the other, $\mathcal{O}_B$ is considered to rule in favor of the side that paid its fees.*
      - *If neither side pays its fees, the previous ruling stands.*
  - *Take $\mathcal{C}_1 = \mathcal{C}_0$.*
  - *While $\mathcal{C}_1 \neq \emptyset$*
    - *If $\#\mathcal{C}_1$ is odd, calculate $m = median(\mathcal{C}_1) \in \mathcal{C}_1$. If $\#\mathcal{C}_1$ is even, choose one of the two middle-most values of $\mathcal{C}_1$ as m in some predictable way (such as by always taking the value on the left).*
    - *Eliminate all l_i and u_i that are on the wrong side of what $\mathcal{O}_B$ decided with respect to m (taking into account the final outcome after any appeals) from $\mathcal{L}$ and $\mathcal{U}$.*
    - *Add m to $\mathcal{L}$ if $\mathcal{O}_B$ has ruled that the true value is higher than m, and add m to $\mathcal{U}$ otherwise.*
    - *Recalculate $\mathcal{L}_0$ and $\mathcal{U}_0$ based on the updated $\mathcal{L}$ and $\mathcal{U}$.*
    - *(Re)calculate $\mathcal{C}_1$ as*

      $$\{median(l_i, u_j) : l_i \in \mathcal{L}_0,\ u_j \in \mathcal{U}_0,\ u_j \leq l_i,\ \nexists\ l_k \in [u_j, l_i) \cap \mathcal{L},\ \nexists\ u_k \in (u_j, l_i] \cap \mathcal{U}\}.$$

Output the average of the largest remaining element of $\mathcal{L}$ and the smallest remaining element of $\mathcal{U}$. Payments are made to respondents according to a structure that will be described in Section 4.

The respondent $\mathcal{USR}_i$ is ruled incorrect and loses his deposit if the final output of Algorithm 1, $v_{\text{output}} \notin (l_i, u_i)$. See Section 4 for details on the payment made to a respondent $\mathcal{USR}_i$ for whom $v_{\text{output}} \in (l_i, u_i)$.

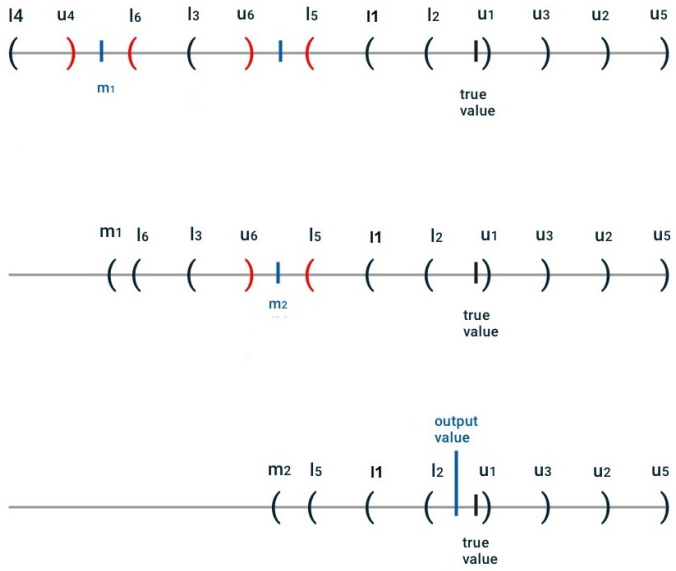

Figure 1 An example of Algorithm 1. Six respondents $\mathcal{USR}_i$ each submit (l_i, u_i). The elements of $\mathcal{C}_0$ are marked by the short, straight blue lines. The binary oracle $\mathcal{O}_B$ is called to rule on each element of $\mathcal{C}_0$ in parallel, and then these results are translated into the output via two rounds of the while loop. The three lines show the state of $\mathcal{L} \cup \mathcal{U}$ before the first round, between the two rounds, and after the second round respectively. The execution shown corresponds to $\mathcal{O}_B$ ruling that the true value v is such that $v > m_1$ in round one and $v > m_2$ in round two.

We will choose the respondent deposits to be large enough so that the deposits of respondents who are ruled incorrect is sufficient to cover the fees required for the initial ruling (i.e. excluding potential appeals) of each call of $\mathcal{O}_B$ required by the for loop, namely at every point in $\mathcal{C}_0$. This will require that the deposit D from each respondent be greater than or equal to A, the total amount of fees required by a single, non-appealed call of $\mathcal{O}_B$. In contrast, the fees for any appeals are submitted independently from the respondent deposits (depending on the structure of the cryptoeconomic game used by $\mathcal{O}_B$, this will typically be done by parties interested in winning the stake of the other side as discussed in Section 3.1).

▶ **Proposition 1.** *Enough fees are paid by respondents who are ultimately ruled incorrect to cover the initial round of all required calls to the binary oracle. Specifically,*

$$\#\text{ submissions ruled incoherent} \geq \#\mathcal{C}_0 = \#\text{ rulings required}$$

Proof. As each respondent pays a deposit that includes A, the cost of a call to $\mathcal{O}_B$ before appeals, there is $A \cdot (\#$ submissions ruled incoherent$)$ available to cover the fees of the total initial round of binary oracle calls. By construction, there is a ruling with respect to each point in $\mathcal{C}_0$. So it suffices to prove the first inequality.

Take $c \in \mathcal{C}_0$ such that $c \geq v$, where v is the ultimate output of the oracle. For each such c there are some $l_i \in \mathcal{L}_0$, $u_j \in \mathcal{U}_0$ such that $u_j \leq c \leq l_i$ and there is no $l_k \in (u_j, l_i) \cap \mathcal{L}$. Then l_i was the lower bound of an incoherent submission.

We claim that this process produces a distinct l_i for each $c \in \mathcal{C}_0$. Indeed, if c_1, $c_2 \in \mathcal{C}_0$ are as above with $u_{j,1} \leq c_1 \leq l_{i,1}$ and $u_{j,2} \leq c_2 \leq l_{i,2}$ but $l_{i,1} = l_{i,2}$, then either

- $u_{j,1} \in (u_{j,2}, l_{i,2}]$ which contradicts the definition of $\mathcal{C}_0$
- $u_{j,2} \in (u_{j,1}, l_{i,1}]$ which again contradicts the definition of $\mathcal{C}_0$ or
- $u_{j,1} = u_{j,2}$ which, as $l_{i,1} = l_{i,2}$, implies that $c_1 = c_2$.

Similarly, for each element of $\mathcal{C}_0$ less than v there corresponds some $u_j \in \mathcal{U}_0$ that is the upper bound of an incoherent submission. ◀

We see that all of the rulings of the binary oracle that are needed to evaluate the while loop were, in fact, decided.

▶ **Proposition 2.** *During the while loop, each $\mathcal{C}_1$ that is computed is a subset of $\mathcal{C}_0$. Hence, for each m computed during the loop, $\mathcal{O}_B$ will have ruled either*

- *desired value $\leq m$ or*
- *desired value $> m$*

Proof. The sets $\mathcal{L}$ and $\mathcal{U}$ are only modified during the while loop. There, if $\mathcal{O}_B$ rules that desired value $\leq m$, all elements greater than or equal to m are eliminated from $\mathcal{L}$ and $\mathcal{U}$, and m is added to $\mathcal{U}$. Due to the local way that $\mathcal{C}_1$ is defined, as the elements of $\mathcal{L}$ and $\mathcal{U}$ less than m remain unchanged, the only way a new element could be added to $\mathcal{C}_1$ is if there existed some $l_i \in \mathcal{L}_0$, $m \leq l_i$ such that median$(l_i, m) \in \mathcal{C}_1$. However, when computing $\mathcal{C}_1$, m is a strict upper bound for $\mathcal{L}$, so there will not exist any such l_i. A similar argument applies if $\mathcal{O}_B$ had ruled that desired value $> m$. ◀

▶ Remark 3. Note that is possible that $\mathcal{O}_B$'s rulings on different points in $\mathcal{C}_0$ will be incoherent; e.g. one call of $\mathcal{O}_B$ will rule that $v \leq m_1$ and another call of $\mathcal{O}_B$ will rule that $v > m_2$ even if $m_2 \geq m_1$. This does not prevent the algorithm from halting as the while loop gives priority to the decisions required along the path of a binary search.

▶ Remark 4. At the expense of additional gas, after each appeal round in algorithm 1, one can test whether the required $\mathcal{O}_B$ calls for the while loop to terminate have been finalized, i.e. have not been appealed. Depending on how underlying binary oracle it structured, it may be necessary to resolve all of the appealed calls of $\mathcal{O}_B$ for an appropriate payment of its internal incentives, however this need not unnecessarily delay the finalization of the result of Algorithm 1.

A priori, it is conceivable that at some point of Algorithm 1, $\mathcal{L}$ and $\mathcal{U}$ become empty and the last step of the algorithm fails. We see that this cannot, in fact, occur.

▶ **Proposition 5.** *Suppose that there is at least one submission (l_*, u_*) to the oracle. Then if Algorithm 1 halts, it returns a value in $\mathcal{S}$.*

Proof. We will see that neither $\mathcal{L}$ nor $\mathcal{U}$ becomes the empty set. Then, in particular, the last step of Algorithm 1 is well-defined because there *is* a largest remaining element in $\mathcal{L}$ and a smallest remaining element of $\mathcal{U}$.

We argue inductively that $l = \min\{\mathcal{L} \cup \mathcal{U}\}$ is in $\mathcal{L}$ after each round of the while loop. Before the first round, this is clearly true, as we have assumed there is at least one submission, and for each submitted element of $\mathcal{U}$ there is a corresponding, smaller element of $\mathcal{L}$. In each round of the while loop, after the appropriate value of m is calculated, either

- $\mathcal{O}_B$ rules that desired value $> m$ or
- $\mathcal{O}_B$ rules that desired value $\leq m$.

In the first case, m is added to $\mathcal{L}$, and it also becomes $\min\{\mathcal{L} \cup \mathcal{U}\}$. In the second case, $\min\{\mathcal{L} \cup \mathcal{U}\}$ is left unchanged, and hence in $\mathcal{L}$. Hence $\mathcal{L}$ never becomes the empty set. A similar argument shows $\mathcal{U} \neq \emptyset$. ◀

Moreover, thinking of $\mathcal{C}_1$ as a set of inconsistencies that prevents the oracle from finding an answer that is a consensus among the different responses, we see that the number of these inconsistencies is reduced by half after each round of the while loop.

▶ **Lemma 6.** *Each round of the while loop reduces the length of $\mathcal{C}_1$ by at least half.*

Proof. Consider a given round of the while loop. Denote $n = \#\mathcal{C}_1$ in this round. Note that the addition of m to $\mathcal{L}$ or $\mathcal{U}$ after the call to $\mathcal{O}_B$ cannot create any new elements of $\mathcal{C}_1$, as m becomes either the smallest element of $\mathcal{L}$ or the largest element of $\mathcal{U}$.

If n is odd, then $\frac{n-1}{2}$ elements in $\mathcal{C}_1$ are on either side of m. Hence at least this many elements are eliminated, whichever way $\mathcal{O}_B$ rules. Moreover, m itself is given as the median of an upper bound u_i to the left and a lower bound l_j to the right. Depending on the ruling of $\mathcal{O}_B$, either u_i or l_j is eliminated. So m is also removed from $\mathcal{C}_1$.

If $n = 2k$ is even, then one of the two most central elements of $\mathcal{C}_1$ is used as m. Thus, on one side of m there will be $k-1$ elements of $\mathcal{C}_1$ and on the other k. Hence there are at least k elements that are eliminated by the choice of $\mathcal{O}_B$, including m itself which again is also removed from $\mathcal{C}_1$. ◀

Lemma 6 has the immediate consequence that:

▶ **Corollary 7.** *Algorithm 1 halts.*

In Section 6, we will consider more precise estimates on the complexity of Algorithm 1.

4 Incentivizing respondents to submit short intervals

In this section we will describe what payouts are made to respondents based on the results of Algorithm 1. For the moment, as discussed in Section 3.2, we imagine that there is no up-front cost paid by the party that sets up whatever application that requires the oracle of Algorithm 1. Indeed, the cost of any initial round (excluding appeals) calls to $\mathcal{O}_B$ that are required in the execution of Algorithm 1 is paid by the respondents. While appeal costs may be covered by other parties (such as the fee insurers imagined by Kleros [1]), one would expect that most appeal fees would also be covered by respondents. Then, while some respondents may make gains from paying fees for the position that is ultimately coherent and winning stake from respondents who lose their deposits, the amount that must be paid to $\mathcal{O}_B$ will collectively mostly come from the respondents. Thus, in terms of the internal incentives of Algorithm 1, the respondents are playing a negative sum game, and in particular are not, on average, compensated for their efforts. In the event that there are many parties with an interest in the result of the oracle, it is nonetheless not unreasonable to expect submissions from respondents in this setting. Effective models for having an "Asker" that pays a fee which can cover compensation for respondents is a potential subject for future work.

Recall, a respondent places a deposit D which he loses if the interval he submits (l_i, u_i) does not contain the output of Algorithm 1. A priori a user might then want to submit a very large interval such as $(-\infty, \infty)$ in order to be guaranteed to be correct. (Note again, parties with an external financial interest in the result of the oracle may nonetheless want an incentive to submit useful estimates). We will design the redistribution mechanism so that respondents have an incentive to submit more precise estimates. To do this we will weight their payouts by an inverse exponential of the length of the submitted intervals.

Consider a respondent $\mathcal{USR}_i$ who submits an interval $I_i = (l_i, u_i)$. If $\mathcal{S}$ is a metric space, such as $\mathbb{R}$, one can take $\text{length}(I_i) = u_i - l_i$ (or more generally as the distance from l_i to u_i). Then, if the ultimate response to the oracle is not in I_i, the user loses his deposit D. If the response is in I_i, the user receives

$$\frac{\#\text{ incorrect responses} \cdot D - \text{cost of first round } \mathcal{O}_B \text{ fees}}{\sum_{j \text{ such that } \mathcal{USR}_j \text{ correct}} \alpha^{-\text{length}(I_j)}} \cdot \alpha^{-\text{length}(I_i)}, \tag{1}$$

where $\alpha > 1$ is some fixed constant. Note that this quantity is positive by Proposition 1. If $\mathcal{S}$ is not a metric space, then the payoff can be split equally between correct respondents.

As such, the sum of the lost deposits from the incorrect users is equal to the sum of the payouts to the correct respondents plus the amount required to pay the fees required by $\mathcal{O}_B$ in the first round of each call. Notice that if a respondent submits an infinite length interval such as $I_i = (-\infty, \infty)$, then $\alpha^{-\text{length}(I_i)} = 0$, for payoffs given by formula 1. So the respondent obtains no reward but suffers no penalty. On the hand, respondents who submit more precise intervals obtain higher rewards.

In Section 8 we will analyze respondent behavior under the incentives given by formula 1. We will show that, under hypotheses on α and certain heuristic assumptions about the behavior of $\mathcal{O}_B$, that it is also not profitable for respondents to submit intervals that are smaller than the precision with which a respondent could reasonably know the desired value. In general, it would be an interesting subject for future experiments to determine how tweaking the weighting of these payoffs, particularly the constant α, would change user behavior to encourage them to submit precise answers for high payoffs, accepting the risk that a very precise answer risks being ruled incorrect even if answered in good faith.

5 Bounds on time griefing in terms of attacker resources

In this section, we estimate an attacker's ability to delay the oracle as a function of her resources. First note,

▶ **Proposition 8.** *If all intervals submitted by the respondents are correct, that is to say the true value $v \in I_i$ for all i, then no calls to $\mathcal{O}_B$ are required.*

Proof. If $v \in I_i = (l_i, u_i)$ for all i, then $l_i < v$ for all v. Similarly $v < u_j$ for all j. Hence $\#\mathcal{C}_0 = \emptyset$ and no calls to $\mathcal{O}_B$ are made. ◀

Now we consider situations where we have an attacker. All of the calls of $\mathcal{O}_B$ of the for loop are performed in parallel, however an attacker can attempt to delay the result of the oracle by appealing one or more of these decisions. As discussed in Section 3.1, each appeal round of $\mathcal{O}_B$ takes t time and all other operations take negligible time. Again, we take that the fees for an initial round ruling of $\mathcal{O}_B$ to be A.

Suppose that appeal fees are such that

$$\min\left\{\sum_{j=1}^{i} f_{\mathcal{A},j}, \sum_{j=1}^{i} f_{\mathcal{B},j}\right\} \geq K \cdot A \cdot 2^i,$$

for all i, where $K > 0$ is a constant that depends only on what algorithm is used for the underlying binary oracle. This is particularly the case if one uses either Kleros [15] or Augur [17] as the binary oracle.[1] Denote by R the attacker's financial resources.

▶ **Proposition 9.** *Suppose that all responses submitted other than the attacker's are ruled correct, namely the output value is in the submitted interval, and suppose that $f_{\mathcal{A},i}$ and $f_{\mathcal{B},i}$ are as above. Then, the maximum number of appeals required is $O_A(\log_2(R))$, and hence the maximum amount of time an attacker can delay a result is $O_A(t \cdot \log_2(R))$, where the implicit constants are allowed to depend on A.*

[1] This is explicitly the case for Kleros [1]. In Augur [17], applied to a binary decision between $\mathcal{A}$ and $\mathcal{B}$, the dispute bond required to dispute a pending outcome of $\mathcal{B}$ is $2S(\mathcal{B}, n) - S(\mathcal{A}, n) \geq S(\mathcal{B}, n) \geq S(\mathcal{A}, n)$, where $S(*, n)$ denotes the total amount of REP staked on choice $*$ prior to the nth round. Hence, $f_{\mathcal{A},i} \geq \sum_{j=1}^{i-1} f_{\mathcal{A},j}$ $\forall i$. Then the bound follows from a standard induction argument.

Proof. If the attacker repeatedly appeals a given decision, then the appeal fees through the mth appeal are at least $K \cdot A \cdot 2^m$. Then, even if the attacker uses all of her resources in appealing a single decision, we have

$$K \cdot A \cdot 2^m \leq \text{money spent by Attacker} \leq R \Rightarrow m \leq \log_2\left(\frac{R}{K \cdot A}\right) = O_A(\log_2 R).$$

(Note that as $\mathcal{O}_B$ is assumed to be always correct, the attacker will ultimately lose her appeal so she these resources will, in fact, be consumed.) ◀

6 Running time and number of calls to binary oracle

In Section 5, we examined an attacker's ability to delay the execution of the oracle by forcing additional appeals. In this section, we will examine the effect of one or more incoherent respondents on the running times of the for and while loops. This is particularly relevant in evaluating the gas costs of Algorithm 1.

Recall, the cost of submitting an incorrect response (l_{attack}, u_{attack}) is D, via a lost deposit when it is eventually determined that the ultimate answer to the oracle is not in the submitted interval. Once again, we denote by R the collective financial resources of respondents who submit intervals that are ultimately ruled to be incoherent, which without loss of generality we can assume to all be controlled by a single attacker.

▶ **Proposition 10.** *Suppose that all responses submitted other than the attackers are ruled correct, namely the output value is in each of these intervals. Then there are at most* $\frac{R}{D}$ *many calls to* $\mathcal{O}_B$ *that must be resolved during the for loop. Consequently, the while loop requires at most* $\max\{\log_2(\#\mathcal{C}_0) + 1, 0\} \leq \max\left\{\log_2\left(\frac{R}{D}\right) + 1, 0\right\}$ *rounds.*

Proof. The attacker can only place at most $\frac{R}{D}$ incorrect solutions. So, by Proposition 1,

$$\#\text{ rulings required} = \#\mathcal{C}_0 \leq R/D.$$

Then, by Lemma 6, it is sufficient to have k many rounds of the while loop such that $\left(\frac{1}{2}\right)^k \#\mathcal{C}_0 \leq \left(\frac{1}{2}\right)^k \frac{R}{D} < 1$. Hence, it is sufficient to have $k = \max\left\{\log_2\left(\frac{R}{D}\right) + 1, 0\right\}$ rounds. ◀

Then, if Algorithm 1 is implemented in such a way that $\mathcal{L}$ and $\mathcal{U}$ are pre-sorted (by submitters including the indices of their entry in the lists), the total on-chain running time of Algorithm 1 is $O\left(\#\text{submissions} \cdot \left(1 + \max\left\{\log_2\left(\frac{R}{D}\right) + 1, 0\right\}\right)\right)$, plus at most $\frac{R}{D}$ calls to $\mathcal{O}_B$.

7 User-calibrated precision

In this section, under idealized assumptions about the results produced by $\mathcal{O}_B$, we study the precision of the output of Algorithm 1. Particularly, we see that it depends on the lengths of the intervals submitted by the respondents.

▶ **Theorem 11.** *Suppose that the true value that Algorithm 1 is trying to determine is* v*, and that the underlying binary oracle* $\mathcal{O}_B$ *is always correct in determining whether a value is greater or less than* v*. Suppose that some respondent submits the interval* $I = (l_0^*, u_0^*)$ *such that* $v \in I$*. Furthermore, suppose that this respondent is willing to pay any required appeal fees in at most* $2\max\{\log_2(\#\mathcal{C}_0) + 1, 0\}$ *specifically chosen calls of* $\mathcal{O}_B$ *on behalf of claims that would be implied by* $v \in I$*. Then the response output by the oracle is in* I*.*

Proof. Consider the calls of $\mathcal{O}_B$ with respect to the values m_i that are considered through the various rounds of the while loop. Suppose for the moment that the respondent pays any appeal fees required when $m_i \notin I$ for rulings consistent with $v \in I$. Then, suppose we have passed through the while loop k times. We will iteratively define an interval I_k such that

$$\text{ultimate response} \in I_k \subseteq I.$$

We define these intervals as either of the form $I_k = (l_k^*, u_k^*)$ or the form $I_k = (l_k^*, u_k^*]$. We take $I_0 = I = (l_0^*, u_0^*)$ and then for $k > 0$:

$$I_k = \begin{cases} (l_{k-1}^*, m_k] & : \text{ if } \mathcal{O}_B \text{ rules that } v \leq m_k,\ m_k < u_{k-1}^* \\ I_{k-1} & : \text{ if } \mathcal{O}_B \text{ rules } v \leq m_k,\ m_k \geq u_{k-1}^* \\ (m_k, u_{k-1}^*) & : \text{ if } \mathcal{O}_B \text{ rules that } v > m_k,\ m_k \geq l_{k-1}^*,\ I_{k-1}^* = (l_{k-1}^*, u_{k-1}^*) \\ (m_k, u_{k-1}^*] & : \text{ if } \mathcal{O}_B \text{ rules that } v > m_k,\ m_k \geq l_{k-1}^*,\ I_{k-1}^* = (l_{k-1}^*, u_{k-1}^*] \\ I_{k-1} & : \text{ if } \mathcal{O}_B \text{ rules that } v > m_k,\ m_k \leq l_{k-1}^* \end{cases}$$

We note that in the first case, we must, in fact, have $l_{k-1}^* \leq m_k$, so I_k is well-defined. If l_{k-1}^* was a previous round m_i, then it must be a lower bound on $\mathcal{L} \cup \mathcal{U}$ in the kth round. Hence $l_{k-1}^* \leq m_k$. Otherwise, if $m_k < l_{k-1}^* = l_0^*$, this dispute is one in which we have assumed that the respondent is willing to pay appeal fees on behalf of $m_k < l_0^* < v$. However, this is incoherent with $\mathcal{O}_B$ ruling that $v \leq m_k$ by our assumptions on the correctness of $\mathcal{O}_B$. Similarly, we see I_k is, in fact, a non-empty interval in all cases, whose endpoints consist of elements of $\mathcal{L}$ and $\mathcal{U}$ in the kth round. Moreover, each $I_k \subseteq I_{k-1} \subseteq I$ by construction.

As the algorithm halts by Corollary 7, eventually, after w rounds of the while loop, all lower bounds in $\mathcal{L}$ will be (strictly) less than all upper bounds in $\mathcal{U}$ with output satisfying

$$l_j < \text{ultimate response} < u_i, \text{ for all } i, j.$$

In particular, the response is (strictly) between l_w^* and u_w^*, so

$$\text{ultimate response} \in I_w \subseteq I.$$

Finally, we show that there is at most one value in $\mathcal{C}_0$ which can arise as an $m_k \leq l_0^*$ for which the respondent would need to pay appeal fees in the kth round of the while loop. Instead suppose that $c_{k,1}, c_{k,2} \in \mathcal{C}_0$ with $c_{k,1}, c_{k,2} \leq l_0^*$ that could each arise as m_k in different executions. Suppose without loss of generality $c_{k,1} \leq c_{k,2}$. Take c_j to be the last common ancestor of these values in the binary search tree, then $c_{k,1} \leq c_j \leq c_{k,2} \leq l_0^*$. As $c_{k,1}$ and $c_{k,2}$ can arise as m_k, c_j can arise as m_j in its round. Then as the respondent is assumed to pay appeal fees for calls regarding $c_j \leq l_0^* < v$, $\mathcal{O}_B$ must rule that $v > c_j$. Hence $c_{k,1}$ cannot arise as a value of m_k. By Proposition 10 and repeating this argument for values $m_k > u_0^*$, the respondent need only pay appeal fees in at most $2 \max\{\log_2(\#\mathcal{C}_0) + 1, 0\}$ calls. ◀

A consequence of Theorem 11 is that if users require that the oracle output very precise values, they need only submit very precise interval estimates as respondents in which they are nonetheless confident that the true answer lies. Another consequence of Theorem 11, is that, again under idealized assumptions on $\mathcal{O}_B$, honest respondents will never be penalized.

▶ **Corollary 12.** *Suppose that the true value that Algorithm 1 is trying to determine is v, and that $\mathcal{O}_B$ is always correct in determining whether a value is greater or less than v. Suppose that a respondent submits the interval $I = (l_0^*, u_0^*)$ such that $v \in I$. Furthermore, suppose that this respondent is willing to pay all required appeal fees in $2 \max\{\log_2(\#\mathcal{C}_0) + 1, 0\}$ specifically chosen calls of $\mathcal{O}_B$ on behalf of claims that would be implied by $v \in I$. Then the user will not lose any appeal fees or deposits.*

Proof. By Theorem 11, the eventual value output by the procedure will be in I, hence the respondent will not lose his deposit. As discussed in the proof of Theorem 11, in the cases in which the respondent is assumed to contribute appeal fees (if necessary), he takes positions consistent with a true value of v and hence is always on the winning side. ◀

8 Equilibria in the respondent game

A key challenge of price oracles is that it is often unreasonable to speak about the price of an asset as being defined beyond a certain precision. An asset can be traded in many marketplaces simultaneously and while one might average together some weighted version of the prices in these different marketplaces, this will inevitably only give an approximation of the price. As a result, one might argue that there is some interval $(v-\epsilon, v+\epsilon)$ such that any element of this interval could be argued to be the price. Considering the transcendental nature of $\mathbb{R}$, one can expect this phenomenon to hold for other $\mathbb{R}$ or $\mathcal{S}$ valued oracles as well.

In previous sections, we have sometimes taken the idealized hypothesis that $\mathcal{O}_B$ rules "honestly" with respect to whether a given x is higher or lower than some single "true value." This might be a realistic (if optimistic) assumption when $x \notin (v-\epsilon, v+\epsilon)$. However, when presented with $x \in (v-\epsilon, v+\epsilon)$, it is more realistic to consider the choice made by $\mathcal{O}_B$ as being inevitably random, even when $\mathcal{O}_B$ is "honest."

In this section, we will consider a simplified model where the output of Algorithm 1 is distributed uniformly over $(v-\epsilon, v+\epsilon)$, and we will analyze respondent payoffs and incentives. Similar analysis using other distributions may be a subject for future work; however, already this simple model is not completely unreasonable. Heuristically, imagine that $\mathcal{O}_B$ responds "yes" and "no" with equal probability if posed the question "is x greater than the true value" for any $x \in (v-\epsilon, v+\epsilon)$. Further, suppose respondents placed intervals in such a way that $\mathcal{C}_0 \cap (v-\epsilon, v+\epsilon)$ consists of equally spaced points. Then, as the number of such points increases, the output's distribution becomes well approximated as uniform on $(v-\epsilon, v+\epsilon)$.

In this model, we examine conditions under which there is no economic incentive in terms of the payouts to coherent respondents for them to submit intervals smaller than 2ϵ. Of course, respondents may have some external interest in the output of Algorithm 1 such that they are incentivized to submit smaller intervals.

▶ **Proposition 13.** *Consider a model as described above, where the output value of Algorithm 1 is drawn uniformly from $(v-\epsilon, v+\epsilon)$ and payouts to correct respondents are given according to formula 1. There is an equilibrium where all respondents submit responses containing $(v-\epsilon, v+\epsilon)$; hence the game played by the respondents is Bayesian-Nash incentive-compatible. Moreover, suppose $\alpha^{2\epsilon} < e \approx 2.718$ and suppose that all respondents other than $\mathcal{USR}_i$ submit intervals that either include or do not intersect $(v-\epsilon, v+\epsilon)$. Then the respondent $\mathcal{USR}_i$ maximizes his expected payoff by submitting the interval $I_i = (v-\epsilon, v+\epsilon)$.*

Proof. The first claim is clear from the fact that rewards for respondents are paid from the lost deposits of other respondents. Now assume $\alpha^{2\epsilon} < e$. Suppose that a respondent $\mathcal{USR}_i$ submits an interval I_i such that $\text{length}(I_i) = \delta_i$ and $I_i \subseteq (v-\epsilon, v+\epsilon)$. Then, he has a $\frac{\delta_i}{2\epsilon}$ chance of being ruled correct and a $1 - \frac{\delta_i}{2\epsilon}$ chance of being ruled incorrect. Note that, as we assume that all other respondents submit intervals that either include or do not intersect $(v-\epsilon, v+\epsilon)$, the payoff for a response depends only on δ_i and whether $v_{\text{output}} \in I_i$. Hence,

$$E[\text{submit } I_i] = \frac{\#\text{ incorrect responses} \cdot D - \text{cost of first round } \mathcal{O}_B \text{ fees}}{\alpha^{-\delta_i} + \sum_{j \text{ such that } \mathcal{USR}_j \text{ correct}, j \neq i} \alpha^{-\text{length}(I_j)}} \cdot \alpha^{-\delta_i} \cdot \frac{\delta_i}{2\epsilon} - D \cdot \left(1 - \frac{\delta_i}{2\epsilon}\right).$$

However, the payoff for the honest strategy of submitting $I_i = (v - \epsilon, v + \epsilon)$ is given by

$$E[\text{honest}] = \frac{\#\text{ incorrect responses} \cdot D - \text{cost of first round } \mathcal{O}_B \text{ fees}}{\alpha^{-2\epsilon} + \sum_{j \text{ such that } \mathcal{USR}_j \text{ correct}, j \neq i} \alpha^{-\text{length}(I_j)}} \cdot \alpha^{-2\epsilon}.$$

Denote

$$A = \sum_{j \text{ such that } \mathcal{USR}_j \text{ correct}, j \neq i} \alpha^{-\text{length}(I_j)} \geq 0.$$

Then if we have

$$\frac{1}{\alpha^{-\delta_i} + A} \cdot \alpha^{-\delta_i} \cdot \frac{\delta_i}{2\epsilon} - \frac{1}{\alpha^{-2\epsilon} + A} \cdot \alpha^{-2\epsilon} \leq 0$$

$$\Leftrightarrow \left(1 + A\alpha^{2\epsilon}\right) \cdot \frac{\delta_i}{2\epsilon} - (1 + A\alpha^{\delta_i}) \leq 0,$$

that is sufficient to see that the honest strategy yields a higher expected payout.

However, $\frac{\delta_i}{2\epsilon} \in [0, 1]$, so we define

$$f(x) = xA\alpha^{2\epsilon} + x - A\alpha^{2\epsilon x} - 1$$

for $x \in [0, 1]$. Then

$$f'(x) = A\alpha^{2\epsilon} + 1 - A\alpha^{2\epsilon x} \cdot \ln\left(\alpha^{2\epsilon}\right) \geq A\alpha^{2\epsilon} + 1 - A\alpha^{2\epsilon} \cdot \ln\left(\alpha^{2\epsilon}\right) > 0$$

by the assumption that $\alpha^{2\epsilon} < e$. Then as $f(1) = 0$, one has that $f(x) \leq 0$ for all $x \in [0, 1]$. ◀

9 Conclusion

We have presented a completely crowd-sourced oracle for values in smart contracts from dense totally ordered sets that we expect to be particularly applicable as a price oracle. This proposal takes as an ingredient an oracle that can make binary decisions, for which one could use, in particular, the existing systems of Kleros, Augur, or ASTRAEA, then extending the influx of knowledge about the real world that they provide to a wider setting. The number of times the binary oracle must be called is limited to a reasonable bound in terms of the resources of parties who propose incoherent answers, not calling the system at all if all respondents submit mutually consistent answers. Hence the time required to compute this oracle should be suitable for many applications. Furthermore, the precision with which our proposed oracle returns its final answer is tuned to the precision of the most precise correct respondent so that the system can be as precise as its users require it to be.

References

1 Draft: fees in Kleros. https://github.com/kleros/research-docs/commit/e99053d5b81f6c8126352362a1cfe07f331033bd. Consulted: March 2019.

2 Ethereum white paper: A next-generation smart contract and decentralized application platform. https://github.com/ethereum/wiki/wiki/White-Paper. Consulted: April 2018.

3 Gnosis FAQ. https://gnosis.pm/faq. Consulted: April 2018.

4 Gnosis whitepaper. https://gnosis.pm/resources/default/pdf/gnosis-whitepaper-DEC2017.pdf. December 2017.

5 Technical Assessement of Augur. SportCrypt, Medium https://medium.com/@SportCrypt/technical-assessment-of-augur-5ff1a74df327. August 2018.

6 TLSnotary - a mechanism for independently audited https sessions. `https://tlsnotary.org/TLSNotary.pdf`. September 2014.

7 John Adler, Ryan Berryhill, Andreas G. Veneris, Zissis Poulos, Neil Veira, and Anastasia Kastania. ASTRAEA: A decentralized blockchain oracle. *2018 IEEE Confs on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology, Congress on Cybermatics*, 2018.

8 Nadja Beneš. Getting to the Core: An Overview of our Contract Architecture. Gnosis blog `https://blog.gnosis.pm/getting-to-the-core-4db11a31c35f`. August 2017.

9 Vitalik Buterin. Ethereum and oracles. Ethereum Blog, `https://blog.ethereum.org/2014/07/22/ethereum-and-oracles/`. July 2014.

10 Vitalik Buterin. SchellingCoin: A minimal-trust universal data feed. Ethereum Blog: `https://blog.ethereum.org/2014/03/28/schellingcoin-a-minimal-trust-universal-data-feed/`. March 2014.

11 Lin William Cong, Zhiguo He, and Jingtao Zheng. Blockchain Disruption and Smart Contracts. *SSRN Electronic Journal*, January 2017. `doi:10.2139/ssrn.2985764`.

12 Mike Dalton. Chainlink acquires Town Crier, a hardware-based oracle. Unhashed `https://unhashed.com/cryptocurrency-news/chainlink-acquires-town-crier-hardware-based-oracle/`. November 2018.

13 Michael del Castillo. Thomson Reuters to power blockchain contracts with experimental service. Coindesk, `https://www.coindesk.com/thomson-reuters-power-blockchain-contracts-new-experimental-service/`. June 2017.

14 Steve Ellis, Ari Juels, and Sergey Nazarov. ChainLink: A decentralized oracle network. `https://link.smartcontract.com/whitepaper`. September 2017.

15 Clément Lesaege and Federico Ast. Kleros: Short paper v1.0.5. `https://kleros.io/assets/whitepaper.pdf`. January 2018.

16 The Maker Team. The Dai Stablecoin System. `https://makerdao.com/whitepaper/DaiDec17WP.pdf`. December 2017.

17 Jack Peterson and Joseph Krug. Augur: a decentralized, open-source platform for prediction markets. `https://bravenewcoin.com/assets/Whitepapers/Augur-A-Decentralized-Open-Source-Platform-for-Prediction-Markets.pdf`. July 2018.

18 Thomas Schelling. *The Strategy of Conflict*. Harvard University Press, 1980. URL: `https://books.google.ca/books?id=7RkL4Z8Yg5AC`.

19 Paul Sztorc. Truthcoin: Peer-to-peer oracle system and prediction marketplace. `https://www.truthcoin.info/papers/truthcoin-whitepaper.pdf`. Version 1.5, December 2015.

20 Katrin Tinn. Blockchain and the future of optimal financing contracts. In *European Finance Association 45th Meeting*, EFA 2018. European Finance Association (EFA), 2018. `doi:10.2139/ssrn.3072854`.

21 Fan Zhang, Ethan Cecchetti, Kyle Croman, Ari Juels, and Elaine Shi. Town Crier: An authenticated data feed for smart contracts. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 270–282, New York, NY, USA, 2016. ACM. `doi:10.1145/2976749.2978326`.