Draft (April 2019): A Smart Contract Oracle for Approximating Real-world, Real Number Values - Variant

William George and Clément Lesaege

This is an extension to the existing oracle article [1]. In both versions we assume that we have access to an oracle $\mathcal{O}_B$ which can provide answers to binary questions. The existing structure of Kleros can be used as $\mathcal{O}_B$.

Compared to [1], this version has the advantage that respondents need only insure a more manageable number of appeals at a given time in order for their submission to remain viable. On the other hand, more calls to the underlying binary oracle $\mathcal{O}_B$ may be required and the worst case running time is worse.

Take $\mathcal{S}$ to be a dense, totally ordered set (such as $\mathbb{R}$). When $\mathcal{S} \neq \mathbb{R}$ we define the median as in [1].

1 Proposal

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. Denote the set of such intervals by $\mathcal{I}_0$.*

- *Take $resolved = 0$, $t = 0$.*
- *While $resolved = 0$*
  - *Based on the lower and upper bounds of the intervals in $\mathcal{I}_t$, sort the lower bound responses into a list $\mathcal{L}_t$ and the upper bound responses into a list $\mathcal{U}_t$, where in each case identical values are considered as single elements.*
  - *Compute the lists*

$$\mathcal{L}_{t,0} = \{l_i \in \mathcal{L}_t : \exists u_j \in \mathcal{U}_t,\, u_j \leq l_i,\ \nexists\, l_k \in [u_j, l_i) \cap \mathcal{L}_t\}$$

and

$$\mathcal{U}_{t,0} = \{u_i \in \mathcal{U}_t : \exists l_j \in \mathcal{L}_t,\, l_j \geq u_i,\ \nexists\, u_k \in (u_i, l_j] \cap \mathcal{U}_t\}\,.$$

- *Compute*

  $$\mathcal{C}_t = \{median(l_i, u_j) : l_i \in \mathcal{L}_{t,0},\ u_j \in \mathcal{U}_{t,0},\ u_j \leq l_i,\ \nexists\ l_k \in [u_j, l_i) \cap \mathcal{L}_t,\ \nexists\ u_k \in (u_j, l_i] \cap \mathcal{U}_t\}$$

  (So, if we considered $\mathcal{L}_t$ and $\mathcal{U}_t$ in the same line, essentially $\mathcal{L}_{t,0}$ would consist of lower bounds which have an upper bound to their immediate left and $\mathcal{U}_{t,0}$ would consist of upper bounds that have a lower bound to their immediate right. Then $\mathcal{C}_t$ consists of the midpoints between each of these pairs.)

- *If $\mathcal{C}_t = \emptyset$*
  - *Set $resolved = 1$.*
- *If $\mathcal{C}_t \neq \emptyset$*
  - *Take $\mathcal{I}_{t+1} = \emptyset$.*
  - *For each $z \in \mathcal{C}_t$ 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, where here the two sides are as follows:*

      $$desired\ value \leq z$$

      or

      $$desired\ value > z,$$

      where, again, 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, and if neither side pays its fees, the previous ruling stands.
  - *Once appeals are finalized, if $\mathcal{O}_B$ ruled*

    $$desired\ value \leq z,$$

    add $(-\infty, z]$ to $\mathcal{I}_{t+1}$. If $\mathcal{O}_B$ ruled

    $$desired\ value > z,$$

    add (z, ∞) to $\mathcal{I}_{t+1}$.
  - *Increment $t \leftarrow t + 1$.*

- *Set $t_{last} = t$.*
- *Set $l_{t_{last}} = \max\{\mathcal{L}_{t_{last}}\}$, taking $l_{t_{last}} = -\infty$ if $\mathcal{L}_{t_{last}} = \emptyset$.*

- *Set* $u_{t_{last}} = \min\{\mathcal{U}_{t_{last}}\}$*, taking* $u_{t_{last}} = \infty$ *if* $\mathcal{U}_{t_{last}} = \emptyset$*.*
- *For* $s = 1$ *to* $s = t_{last}$
  - *If* $\{l \in \mathcal{L}_{t_{last}-s} : l \geq l_{t_{last}-s+1}, \nexists u \in \mathcal{U}_{t_{last}-s} \cap (l_{t_{last}-s+1}, l)\} \neq \emptyset$*,*
    - *Set* $l_{t_{last}-s} = \max\{l \in \mathcal{L}_{t_{last}-s} : l \geq l_{t_{last}-s+1}, \nexists u \in \mathcal{U}_{t_{last}-s} \cap (l_{t_{last}-s+1}, l)\}$
  - *Otherwise*
    - *Set* $l_{t_{last}-s} = -\infty$
  - *If* $\{u \in \mathcal{U}_{t_{last}-s} : u \leq u_{t_{last}-s+1}, \nexists l \in \mathcal{L}_{t_{last}-s} \cap (u, u_{t_{last}-s+1})\} \neq \emptyset$
    - *Set* $u_{t_{last}-s} = \min\{u \in \mathcal{U}_{t_{last}-s} : u \leq u_{t_{last}-s+1}, \nexists l \in \mathcal{L}_{t_{last}-s} \cap (u, u_{t_{last}-s+1})\}$*,*
  - *Otherwise*
    - *Set* $u_{t_{last}-s} = \infty$*.*

Output median(l_0, u_0)*.*

2 Payout structure

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

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

where $\alpha > 1$ is some fixed constant. Note that this quantity is positive by Proposition 5. If $\mathcal{S}$ is not a metric space, then the payoff can be split equally between correct respondents.

3 Analysis

Proposition 1. *In each round of the while loop,* $\#\mathcal{I}_t$ *is reduced by at least half.*

Proof. At the end of a round of the while loop, $\#\mathcal{I}_t \leq \#\mathcal{C}_t$. Moreover, all elements of $\mathcal{I}_t$ are either of the form $(-\infty, c_0)$ or (c_0, ∞) for some c_0 in the $\mathcal{C}_t$ calculated in that round. By the definition of $\mathcal{C}_t$, two such elements can only contribute an element to the $\mathcal{C}_{t+1}$ if one is of the form $(-\infty, c_0)$ and the other is of the form (c_0, ∞). Moreover, each interval in $\mathcal{I}_t$ can only contribute to a single element of $\mathcal{C}_t$. Then, as

$$\min\{\#\text{elements of form } (-\infty, c_0), \#\text{elements of form } (c_0, \infty)\} \leq \frac{\#\mathcal{I}_t}{2},$$

there can be at most $\frac{\#\mathcal{I}_t}{2}$ elements in $\mathcal{C}_{t+1}$. □

Corollary 1. *Algorithm 1 halts.*

Proposition 2. $l_t \geq l_{t+1}$ *and* $u_t \leq u_{t+1}$ *for all* $t = 0, \ldots, t_{last}$.

Proof. We show that $l_t \geq l_{t+1}$. The argument to show that $u_t \leq u_{t+1}$ is similar. If $l_{t+1} = -\infty$, then the claim clearly holds. Suppose l_{t+1} is finite. Then, as l_{t+1} is by construction an element of $\mathcal{C}_t$, there exist $l \in \mathcal{L}_t$, $u \in \mathcal{U}_t$ such that $l_{t+1} = \text{median}(l, u)$, $u \leq l$, $[u, l) \cap \mathcal{L}_t = \emptyset$, and $(u, l] \cap \mathcal{U}_t = \emptyset$. Particularly, $l \geq l_{t+1}$ and $\mathcal{U}_t \cap (l_{t+1}, l) = \emptyset$. Hence the set over which l_t is defined is non-empty. Then $l_t \geq l_{t+1}$ by the definition of this set.

□

Proposition 3. $l_t < u_t$ *for all* $t = 0, \ldots, t_{last}$.

Proof. This is true for $t = t_{last}$ as $\max\{\mathcal{L}_{t_{last}}\} < \min\{\mathcal{U}_{t_{last}}\}$ implies the existence of an element in $\mathcal{C}_{t_{last}}$. Furthermore, the cases where one or both of $l_{t_{last}}$ or $u_{t_{last}}$ are infinite are clear. Then the result follows inductively from Proposition 2.

□

Proposition 4. $(l_t, u_t) \cap \mathcal{C}_t = \emptyset$ *for all* $t = 1, \ldots, t_{last}$.

Proof. This is true for $t = t_{last}$ as otherwise the algorithm would not resolve.

Then, suppose that $(l_{t+1}, u_{t+1}) \cap C_{t+1} = \emptyset$ but there exists some $c \in (l_t, u_t) \cap C_t$ for some $t < t_{last}$. By Proposition 2,

$$l_{t+1} \leq l_t < c < u_t \leq u_{t+1}.$$

Then, based on the ruling of $\mathcal{O}_B$, either $c \in \mathcal{L}_{t+1}$ or $c \in \mathcal{U}_{t+1}$. Suppose $c \in \mathcal{L}_{t+1}$. Then, this contradicts the maximum properties of l_{t+1} unless there exists an element $u \in \mathcal{U}_{t+1} \cap (l_{t+1}, c)$. However, $u < c$, $u \in \mathcal{U}_{t+1}$, $c \in \mathcal{L}_{t+1}$ implies the existence of an element in $\mathcal{C}_{t+1} \cap (u, c)$ which is a contradiction. We similarly have a contradiction if $c \in \mathcal{U}_{t+1}$.

□

Proposition 5. *We have*

$$\#\textit{submissions ruled incoherent} \geq \#\mathcal{C}_0.$$

Moreover, suppose each respondent submits a deposit $D \geq 2A$*, where* A *is the required fee for an initial round call to* $\mathcal{O}_B$ *(before any appeals). Then enough fees are paid by respondents who are ultimately ruled incorrect to cover the initial round of all required calls to the binary oracle across all rounds of the while loop.*

Proof. We begin by showing that

$$\#\text{submissions ruled incoherent} \geq \#\mathcal{C}_0.$$

Take $c \in \mathcal{C}_0$ such that $c \geq v_{out}$, where v_{out} 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.

There are at most $\#\mathcal{C}_0$ many calls to $\mathcal{O}_B$ in the first round of the while loop. By Proposition 1, there are at most $2\#\mathcal{C}_0$ total disputes across all rounds of the while loop.

□

Denote by $f_{\mathcal{A},i}$ the fee required in the ith appeal round in support of the position $\mathcal{A}$. Similarly, denote by $f_{\mathcal{B},i}$ the fee required in the ith appeal round in support of the position $\mathcal{B}$.

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. Denote by R the attacker's financial resources. Denote by t the time required by a call to $\mathcal{O}_B$ considering other operations as requiring negligible time.

Proposition 6. *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 the maximum amount of time an attacker can delay a result is

$$O_A\left(t \cdot \log_2(R) \cdot \log_2\left(\frac{R}{D}\right)\right),$$

where the implicit constants are allowed to depend on A.

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.)

Then, by Proposition 1, there are at most $\max\{\log_2(\#\mathcal{C}_0) + 1, 0\}$ many rounds of the while loop, each of which consisting of disputes that can be appealed $O_A(\log_2 R)$ many times. By Proposition 5, placing $\#\mathcal{C}_0$ many ultimately incorrect submissions costs at least $\#\mathcal{C}_0 \cdot D$. Hence, even if the attacker spent all of her resources placing submissions that would ultimately be ruled incorrect, $\#\mathcal{C}_0 \leq \frac{R}{D}$. Then the total time required by this process is

$$O_A\left(t \cdot \log_2(R) \cdot \log_2\left(\frac{R}{D}\right)\right).$$

□

Note - can probably show a slightly more refined version that takes into account an attacker's optimal splitting of resources between appeals and causing extra rounds of while loop - shouldn't be that different.

Proposition 7. *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. Furthermore, the while loop requires at most* $\max\{\log_2(2\#\mathcal{C}_0) + 1, 0\} \leq \max\left\{\log_2\left(2\frac{R}{D}\right) + 1, 0\right\}$ *rounds.*

Proof. The attacker can only place at most $\frac{R}{D}$ incorrect solutions. So, by Proposition 5, $\#\mathcal{C}_0 \leq \frac{R}{D}$ and by Proposition 1, the total number of calls to $\mathcal{O}_B$ is upper bounded by $2\#\mathcal{C}_0 \leq 2\frac{R}{D}$. Then, again by Proposition 1, there are at most $\max\{\log_2(\mathcal{C}_0) + 1, 0\} \leq \max\left\{\log_2\left(\frac{R}{D}\right) + 1, 0\right\}$ many rounds of the while loop.

□

Note that the for loop requires the same number of rounds as the while loop by construction.

Then, if Algorithm 1 is implemented in such a way that $\mathcal{L}$ and $\mathcal{U}$ are presorted (by submitters including the indices of their entry in the lists), the process of computing each $\mathcal{L}_\sqcup$, $\mathcal{U}_t$, l_t, u_t etc takes $O(\#\text{submissions})$ steps. Then the total on-chain running time of Algorithm 1 is

$$O\left(\#\text{submissions} \cdot \left(1 + \max\left\{\log_2\left(\frac{R}{D}\right) + 2, 0\right\}\right)\right),$$

steps plus at most $\frac{2R}{D}$ calls to $\mathcal{O}_B$.

Furthermore, we consider the on-chain storage requirements of Algorithm 1. We must store $\mathcal{L}_t$, $\mathcal{U}_t$, $\mathcal{C}_t$, $\mathcal{I}_t$, l_t, and u_t for all t. Again using our bound on the number of rounds of the while loop, this is a total storage requirement of

$$O(\#\text{submissions} \cdot (1 + \max\{\log_2(R/D) + 1, 0\})).$$

Proposition 8. *Consider the set $\mathcal{I}_t$ at the beginning of some round of the while loop of Algorithm 1. Suppose that there exists some value $v \in I$ for all $I \in \mathcal{I}_t$, then the while loop will resolve in this round and no further calls to $\mathcal{O}_B$ will be required. Particularly, if all intervals submitted by the respondents are coherent, that is to say the value $v \in I$ for all responses I, then no calls to $\mathcal{O}_B$ are required. Also, if in some round of the while loop of Algorithm 1, all rulings of $\mathcal{O}_B$ are coherent with respect to a given value v - i.e. if $\mathcal{O}_B$ rules that*

$$\textit{desired value} \leq z$$

for all $z \geq v$ and that

$$\textit{desired value} > z$$

for all $z < v$ - then Algorithm 1 does not require calls to $\mathcal{O}_B$ in any subsequent rounds of the while loop.

Proof. As $v \in I = (l_i, u_i)$ for all $I \in \mathcal{I}_t$, then $l_i < v$ for all $l_i \in \mathcal{L}_t$. Similarly $v < u_j$ for all $u_j \in \mathcal{U}_j$. Hence $\#\mathcal{C}_t = \emptyset$ and no calls to $\mathcal{O}_B$ are made.

For the last claim, the assumed coherence property implies that v is in all intervals that are added to $\mathcal{I}_{t+1}$, whether of the form $(-\infty, c_0)$ or (c_0, ∞) corresponding to the ruling of $\mathcal{O}_B$.

□

Corollary 2. *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 $I = (l_*, u_*)$ such that $v \in I$. Furthermore, suppose that this respondent is willing to pay all required appeal fees in at most two specifically chosen calls of $\mathcal{O}_B$ per round of the while loop 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 1, 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 1, 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.

□

Theorem 1. *Suppose that the true value 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 $I = (l_*, u_*)$ such that $v \in I$. Furthermore, suppose that the respondent pays any appeal fees required in at most two specifically chosen calls of $\mathcal{O}_B$*

per round of the while loop on behalf of claims that would be implied by $v \in I$. Then the response output by the oracle is in I.

Proof. Suppose that $\mathcal{USR}$ pays any appeal fees required for the c_1, $c_2 \in \mathcal{C}_j$ such that

$$c_1 = \max\{c \in \mathcal{C}_j : c \leq l_*\}$$

and

$$c_2 = \min\{c \in \mathcal{C}_j : c \geq u_*\}$$

in any round j where these quantities are defined.

Note that by Proposition 2, $(l_j, u_j) \subseteq (l_{j+1}, u_{j+1})$ for all $j = 0, \ldots, t_{last} - 1$. Suppose that $(l_*, u_*) \cap (l_0, u_0) = \emptyset$. Then there exists some j_0 such that j_0 is the largest natural number such that $(l_*, u_*) \cap (l_{j_0}, u_{j_0}) = \emptyset$.

A priori, it is possible that $j_0 = t_{last}$, i.e. $(l_*, u_*) \cap (l_j, u_j) = \emptyset$ for all j. This implies $l_* \geq u_{j_0}$ or $u_* \leq l_{j_0}$. Suppose $l_* \geq u_{j_0}$. The case for $u_* \leq l_{j_0}$ is similar. In particular, as users must submit finite values, this implies that $u_{j_0} < \infty$. Then there must exist some $c \in C_{t_{last}-1} \cap (u_{t_{last}}, l_*]$ for which $\mathcal{USR}$ pays the appeal fees and by the assumed correctness of $\mathcal{O}_B$, is ruled in $\mathcal{L}_{t_{last}}$. Otherwise, we would have had $u_{t_{last}} \in \mathcal{L}_{t_{last}}$. Then, $u_{t_{last}} < c$, $u_{t_{last}} \in \mathcal{U}_{t_{last}}$, $c \in \mathcal{L}_{t_{last}}$implies the existence of an element in $\mathcal{C}_{t_{last}} \cap (u_{t_{last}}, c)$. However, $\mathcal{C}_{t_{last}} \neq \emptyset$ contradicts the definition of t_{last}.

Thus, we can assume $j_0 < t_{last}$. Assume again $l_* \geq u_{j_0}$ with the $u_* \leq l_{j_0}$ case being similar. Again this implies that $u_{j_0} < \infty$. Then, again there must be some $c \in C_{j_0-1} \cap (u_{j_0}, l_*]$ for which $\mathcal{USR}$ pays the appeal fees and by the assumed correctness of $\mathcal{O}_B$, is ruled in $\mathcal{L}_{j_0}$. Otherwise, we would have had $u_{j_0} \in \mathcal{L}_{j_0}$. Then $u_{j_0} < c$, $u_{j_0} \in \mathcal{U}_{j_0}$, $c \in \mathcal{L}_{j_0}$ implies the existence of an element $c' \in \mathcal{C}_{j_0} \cap (u_{j_0}, c)$. As j_0 is assumed to be the largest value for which $(l_*, u_*) \cap (l_{j_0}, u_{j_0}) = \emptyset$, we have $l_* < u_{j_0+1}$. Hence, using Proposition 2,

$$l_{j_0+1} \leq l_{j_0} < u_{j_0} < c' \leq l_* < u_{j_0+1}.$$

If c' is ruled in $\mathcal{L}_{j_0+1}$ this contradicts the maximum properties of l_{j_0+1} unless there is an element $u \in \mathcal{U}_{j_0+1} \cap (l_{j_0+1}, c')$, which contradicts Proposition 4. Similarly, if c' is ruled in $\mathcal{U}_{j_0+1}$ this also results in a contradiction. Hence we conclude that $(l_*, u_*) \cap (l_0, u_0) \neq \emptyset$.

Now we show that

$$(l_*, u_*) \cap (l_0, u_0) \neq \emptyset \Rightarrow (l_0, u_0) \subseteq (l_*, u_*).$$

Note that

$$(l_*, u_*) \cap (l_0, u_0) \neq \emptyset$$

implies that there exists a value y such that both $l_* < y < u_*$ and $l_0 < y < u_0$. Suppose $u_0 > u_*$. Note that $\mathcal{U}_0$ consists of respondents' submissions of the form (l, u), $l < u$. Hence $\max\{\mathcal{L}_0 \cup \mathcal{U}_0\} \in \mathcal{U}_0$ and, using Proposition 2, the set defining u_0 cannot be empty. Hence u_0 is finite and in $\mathcal{U}_0$.

Then

$$l_0 < y < u_* < u_0.$$

However, $u_* \in \mathcal{U}_0$, so this contradicts the definition of u_0 unless there is some $l \in \mathcal{L}_0 \cap (u_*, u_0)$. This would imply that there is a value $c \in \mathcal{C}_1 \cap (l_0, u_0) \subseteq \mathcal{C}_1 \cap (l_1, u_1)$ contradicting Proposition 4. Hence, we must have $u_0 \leq u_*$. One sees $l_0 \geq l_*$ similarly.

Then as the output $v_{out} \in (l_0, u_0) \subseteq (l_*, u_*)$, $\mathcal{USR}$ is ultimately ruled as correct.

□

Proposition 9. *Consider a model 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,

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

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

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

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 $x \in [0,1]$.

□

References

[1] William George and Clément Lesaege. A smart contract oracle for approximating real-world, real number values. 2019.