Infochain: A Decentralized, Trustless and Transparent Oracle on Blockchain

Naman Goel[1*], **Cyril van Schreven**[1*], **Aris Filos-Ratsikas**[2] and **Boi Faltings**[1]
[1]Swiss Federal Institute of Technology, Lausanne (EPFL)
[2]University of Liverpool, UK
{naman.goel, boi.faltings}@epfl.ch, cyril.schreven@protonmail.com, aris.filos-ratsikas@liverpool.ac.uk

Abstract

Blockchain based systems allow various kinds of financial transactions to be executed in a decentralized manner. However, these systems often rely on a trusted third party (oracle) to get correct information about the real-world events, which trigger the financial transactions. In this paper, we identify two biggest challenges in building decentralized, *trustless*[1] and transparent oracles. The first challenge is acquiring correct information about the real-world events without relying on a trusted information provider. We show how a peer-consistency incentive mechanism can be used to acquire truthful information from an untrusted and self-interested crowd, even when the crowd has outside incentives to provide wrong informations. The second is a system design and implementation challenge. For the first time, we show how to implement a trustless and transparent oracle in Ethereum. We discuss various non-trivial issues that arise in implementing peer-consistency mechanisms in Ethereum, suggest several optimizations to reduce gas cost and provide empirical analysis.

1 Introduction

With the increasing popularity of the blockchain technology, the implementation of commercial and governmental systems has witnessed a large shift towards distributed and decentralized approaches. In particular, the emergence of the Ethereum platform has given rise to the development of several applications, often referred to as *decentralized apps* or *DAPs*, which aim to apply this latter principle to many areas such as finance, education, intellectual property or government. At the heart of these approaches lies the concept of the *smart contract*, i.e., lines of code that contain the terms of the agreement between the involved parties, which are automatically executed once triggered by events happening in the real world. For example, consider the case of a web service, which is typically dictated by a service level agreement (SLA) between the service provider and the clients. The SLA can be coded into a smart contract between the involved parties which will trigger an automatic payment upon detection of a violation. For instance, if the service guarantees a response time of at most 1 second with high probability, frequent slower responses would trigger automatic compensation. An important issue here is, *how to determine whether the real-world event has actually happened*. In the above example, this means how to determine that the SLA has been violated? We use the case of the web service only as a simple running example but this is in fact a *fundamental challenge in developing information infrastructure for FinTech*.

The need for trusted information about a real-world event that triggers some conditional financial transactions arises in applications ranging from insurance, banking, trade, governance and law etc. The entities responsible for acquiring such data about the real-world events are called *oracles*. Existing solutions include Town Crier and Chainlink among others. Traditionally, oracles are implemented using trusted third party data sources responsible for acquiring the information. However, besides the fact that such an approach is in conflict with the decentralized nature of the blockchain technology, it is also prone to problems such as trustworthiness and cost.

An alternative solution would be to appeal to the "wisdom of the crowds" and ask the users themselves about the information (for e.g., the quality of service received). The idea has also been proposed for outcome resolution in decentralized prediction markets like Augur and Gnosis. While this approach is more decentralized in nature, it poses a significant challenge: the agents can not be relied on to provide correct information. The task of eliciting information from self-interested agents is one of the fundamental problems in game theory, and has been extensively studied. In the center of these investigations lies the literature on peer-consistency mechanisms [Faltings and Radanovic, 2017]; these are game-theoretic mechanisms that incentivize agents to report the information truthfully, even if the information is unverifiable.

In this paper, we propose the employment of peer-consistency mechanisms for the design of trustless, decentralized oracles. This seems like a natural choice, as the usefulness of the oracles is dependent on the quality of the supplied information, which needs to be truthfully elicited from the agents. However, this quest imposes two major challenges:

- For a peer-consistency mechanism to actually work, the

[*] The two authors contributed equally.

[1] Trustless is a term increasingly used in the context of decentralized and blockchain systems meaning *not requiring trust*.

agents must be convinced of its incentive properties, contrary to the traditional case, where the implementation of the mechanism is done by a trusted third-party. *How can one implement the incentive scheme in a transparent and trustless manner, what is the cost and how can we optimize this cost?*

- In many financial settings, agents also have incentives to lie about their true observations and provide false information. In the web service example, the clients would have an incentive to always report "bad" response times, in order for the conditions of the smart contract to be violated in their favor. *How large do the incentives have to be, to counteract the lying incentives, and is the approach economically feasible?*

In this paper, we address the above questions. We summarise our contributions below.

1. We design and implement INFOCHAIN, a completely decentralized peer-consistency based truthful information collection system in Ethereum. We address the following technical challenges in its implementation.
   - Writing data and performing computation on Ethereum's Virtual Machine (EVM) is expensive. Information providers must be compensated for this cost, increasing the overall cost of information acquisition. For the first time, we discuss several non-trivial ways of implementing three different peer-consistency mechanisms in Solidity (Ethereum's programming language) and empirically compare their costs.
   - While transparency is a desired inherent feature of blockchain, the peer-consistency mechanisms are compromised if an agent can see the information submitted by their "peers" before submitting their own information. We propose to use a *commit-reveal protocol* to address this challenge.
   - In order to reduce computation complexity, peer-consistency mechanisms use only one (or a few) randomly selected peer(s) for every agent. However, if the random peer(s) can be predicted, the agents get an opportunity to collude and the mechanisms can be compromised, and this risk is increased by the transparency of blockchain. We show that under reasonable assumptions, random peer selection can still be implemented safely.
2. We analyze the settings when agents have outside incentives to lie. We formally show that even in the presence of such outside incentives peer-consistency can be used to elicit the truth by choosing an appropriate constant to scale the rewards. We show that normally the payments required are a small fraction of the outside incentive.

1.1 Related Work

Many decentralized systems have been proposed for crowdsourcing and information trading [An *et al.*, 2019; Xiong and Xiong, 2019; Lu *et al.*, 2018; Li *et al.*, 2018] but none addresses the issue of providing quality based incentives for information. In a recent and independent work, [Kong *et al.*, 2019] also use peer-consistency for *trust-free* data trading systems but the analysis in this theoretical paper focuses on a secure multi-party computation protocol for rewarding information that loses value if revealed. [Adler *et al.*, 2018] propose a system for a decentralized oracle but it requires a "random assignment" of questions to agents, which has the drawback that agents may be asked to answer questions that they may have no information about.

The literature on peer-consistency is long and extensive [Miller *et al.*, 2005; Prelec, 2004; Waggoner and Chen, 2014; Gao *et al.*, 2016; Agarwal *et al.*, 2017; Liu and Chen, 2017; Goel and Faltings, 2019a; 2019b].; we refer the reader to [Faltings and Radanovic, 2017] for a detailed exposition of the main results. Most relevant to ours are the results related to incentives which are dependent on the outcome, e.g., [Chakraborty and Das, 2016; Chen *et al.*, 2011; Freeman *et al.*, 2017]. These differ from our theoretical contributions in the fact that they only apply to the specific domain of prediction markets.

2 Infochain

To collect truthful information from self-interested agents, we propose a completely decentralized, transparent and trustless system called Infochain. Infochain enables information requesters to post questions, which can be selected by information providers (agents). The questions can be, for example, of the following form: "Is the `responseTime` of web service W less than 1 second?". Once the agents submit information for the questions they select to answer, their payments in Ether are processed by a smart contract. All the collected information and payments are stored on a public blockchain to ensure transparency and immutability.

Peer-consistency. A crucial step in eliciting trustworthy information from self-interested agents is aligning their incentives with honest behavior. Unfortunately, a naive incentive mechanism may invite free riders who submit random information. Designing truthful incentive mechanisms is a hard problem when there is no way to verify the correctness of the information. This issue has been addressed by game theoretic peer-consistency mechanisms [Faltings and Radanovic, 2017]. The broad idea in these mechanisms is to reward the agents by "matching" the information provided by multiple agents, while discouraging any collusion. The state of the art peer-consistency mechanisms guarantee that truth-telling strategy is the highest paying equilibrium and other equilibria are less profitable. We consider three fundamental peer-consistency mechanisms in this paper.

1. **The Output Agreement (OA) Mechanism [Waggoner and Chen, 2014]:** This is perhaps the simplest of all peer-consistency mechanisms. In the OA mechanism, an agent gets a reward of 1 unit only if her answer for a question matches the answer of her peer for the same question. The reward of the agent for a question is the average over the rewards earned by matching with all peers. The final reward of the agent is the average of her rewards from all the questions answered by her.
2. **The Dasgupta and Ghosh (DG) Mechanism [Dasgupta and Ghosh, 2013]:** In the DG mechanism, an

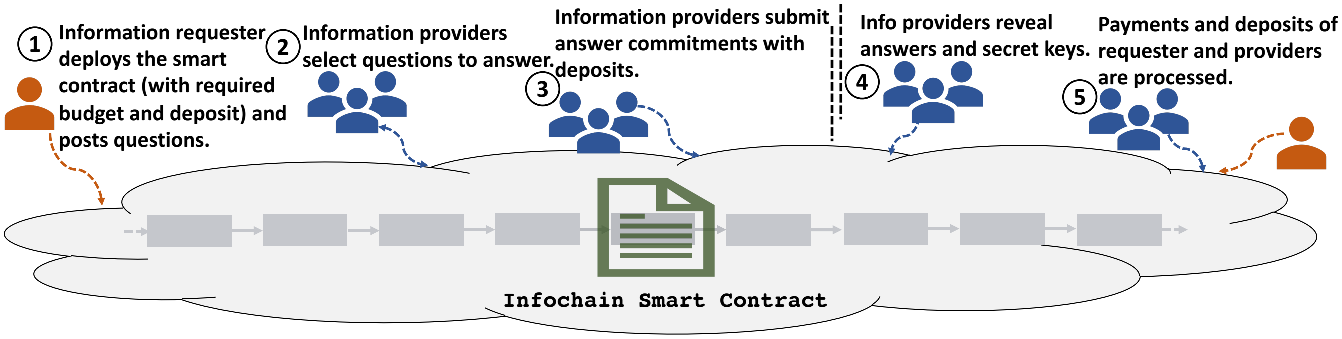

1 Information requester deploys the smart contract (with required budget and deposit) and posts questions.
2 Information providers select questions to answer.
3 Information providers submit answer commitments with deposits.
4 Info providers reveal answers and secret keys.
5 Payments and deposits of requester and providers are processed.
Infochain Smart Contract

Figure 1: Infochain Overview

agent gets a reward of 1 unit if her answer for a question matches the answer of her peer for the same question but also gets a penalty of 1 unit if her answers match the answers of the peer on non-common questions. The DG mechanism requires that two agents, who are peers of one another, must also have some non-common questions that are answered by one of them but not by both. The final reward is calculated by averaging as described in the OA mechanism. The **Correlated Agreement** mechanism [Shnayder *et al.*, 2016] is a generalization of the DG mechanism and exhibits similar computations.

3. **The Peer Truth Serum for Crowdsourcing(PTSC) [Radanovic *et al.*, 2016]:** In PTSC, the reward of an agent i for a question is calculated using the following formula:

$$\begin{cases} \alpha \cdot \left(\frac{\mathbb{1}_{y=y'}}{R_i(y)} - 1 \right) & \text{if } R_i(y) \neq 0 \\ 0 & \text{if } R_i(y) = 0 \end{cases}$$

where y is the answer submitted by the agent and y' is the answer submitted by her peer for the same question. α is a strictly positive scaling constant. $R_i(y) = \text{num}_i(y) / \sum_{\bar{y} \in \{0,1\}} \text{num}_i(\bar{y})$, where $\text{num}_i(y)$ is a function that counts occurrences of y in the answers of all agents (except i) across all questions. The final reward is calculated by averaging discussed earlier.

Traditionally, these mechanisms are implemented by a centralized trusted third party. Implementing them in Infochain, which doesn't assume any centralization or trust, is challenging. In the following subsections, we address the main implementation and theoretical challenges. An overview of Infochain is provided in Figure 1.

2.1 Commit-Reveal Protocol

Transparency is an inherent feature of blockchain. Thus, all the information submitted by an agent is visible to all others. The peer-consistency mechanisms guarantee their incentive compatibility assuming that an agent can only form a belief about what her peers are going to report but doesn't know the actual report of peers. We ensure this in Infochain by making the agents follow a commit-reveal protocol:

1. **Commit:** An agent writes her commitment $keccak256(y, k)$ on the chain, where y is the agent's answer for a given question and k is her secret key.
2. **Reveal:** Once all agents who have selected a question, have finished submitting their commitments for the question or the commitment phase expires, they can reveal their respective secret keys and answers. If the commitment of an agent matches her revealed answer, the answer is written on the chain, otherwise it is discarded.

2.2 Cost Optimizations

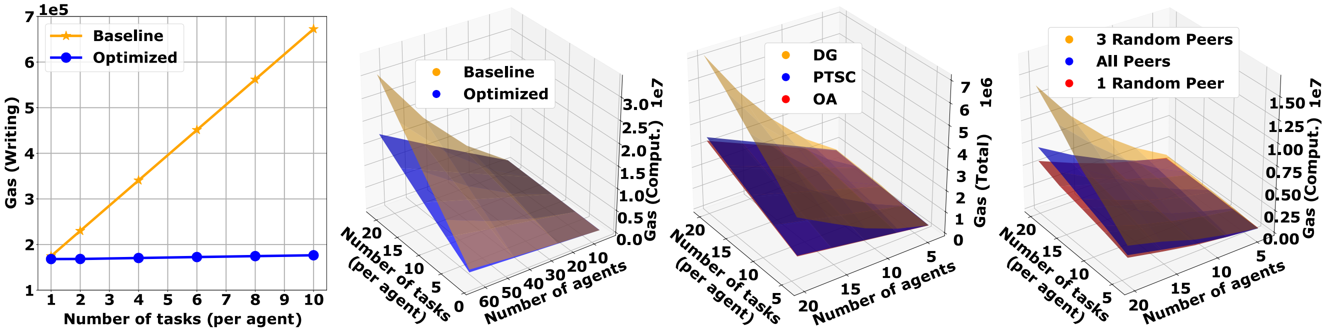
Performing computations on Ethereum's Virtual Machine (EVM) remains an expensive affair. Computation costs on EVM are roughly 10^8 times higher than AWS[2]. [Ryan, 2017] provides a good summary about the costs of basic arithmetic operations and writing operations for different data types. Agents who provide information must be compensated for this cost, increasing the overall cost of information acquisition. We discuss below several non-trivial ways of implementing three different peer-consistency mechanisms in Solidity so that the costs can be minimized.

1. **Optimizing Writing Cost**: To minimize the costs of writing on the chain, agents on Infochain combine multiple answers in the form of a bit vector. This is motivated by two observations. First, the answers are revealed simultaneously and thus, they do not require separate commitments. Second, the EVM operates on 256 bit words, thus a single bit vector is much cheaper to write than other formats.

   Proposition 1. *With the above scheme, each 256-bit commitment can contain up to 42 answers.*

   Proof. Given a hash function $\mathcal{H}$ with a $3k$ bit output, to commit the k bit message m, Alice generates a random k bit string $\mathcal{S}$ and sends Bob $\mathcal{H}(\mathcal{S}||m)$. The probability that any $\mathcal{S}'$, m' exist where $m' \neq m$ such that $\mathcal{H}(\mathcal{S}'||m') = \mathcal{H}(\mathcal{S}||m)$ is $\approx 2^{-k}$. The size of the message sent is limited to one third the size of the output of the hashing function, thus 85 bits. Each answer requires 2 bits: the first determines if the question was answered and the second is the answer. Therefore each commitment can contain 42 answers. □

[2] https://aws.amazon.com/blockchain/



(a) Writing Cost Optimization (b) Comput. Cost Optimization (c) Comparing Mechanisms (d) Random Peer Selection (DG)

Figure 2: Experimental Results

This optimization helps both commit and reveal phases.

2. **Optimizing Computation Cost**: To reduce the cost of computing the rewards, a set of so-called intermediary values is introduced. These values naturally appear at intermediary states of reward computation. They will be precomputed and reused for each agent. What these intermediary values are, depends on the peer-consistency mechanism. This approach allows for the computation to traverse the data a minimum number of times. Since all rewards are computed at the same time, these intermediary values don't need to be written on the blockchain and can be kept in memory instead.

   For an example, consider the PTSC mechanism, which requires relative frequency $R_i(y)$ of the value y while excluding the answer given by agent i. This quantity need not be calculated from scratch for every agent or when every new answer is submitted and neither it is required to be written on the chain. The intermediary values (for e.g. running average) can be kept in memory and used to calculate or update $R_i(y)$ as required.

2.3 Random Peer Selection

In peer-consistency, we can use only one or a few randomly selected peers for reward calculation instead of all peers. This is because, in expectation, the rewards of the agents remain unchanged and thus, the mechanisms with randomly selected peers also offer the same incentive compatibility (except that the variance in rewards increases). This is an interesting tradeoff between computation cost and variance in rewards. However, random peer selection on blockchain is subtle mainly due to the fact that nothing on the chain is a "secret", including the seed for random number generation. If random peers can be known in advance, it may increase the risk of collusion between the agents compromising the incentive compatibility of the mechanisms. In Infochain, we use the block timestamps as well as the mining difficulty level as the seed. This avoids using any trusted third party for random peer selection. The approach works under the assumption that the miners will not try to cheat the smart-contract, which is a reasonable assumption given that the miners have no incentive to do so since they risk losing their mining rewards. The assumption can be violated in extreme scenarios where the financial activity on Infochain (for e.g. the incentive amounts) exceed the mining rewards.

2.4 Negative Payments

The DG mechanism and the PTSC allow negative payments, which is implemented in Infochain by making agents submit refundable deposits. Information requesters also deposit the payment budget and an additional refundable deposit. Any outstanding deposits of the agents and the requester are returned after the payments and computations costs are settled.

3 Experiments

We now discuss the results of some experiments performed on Infochain. The performance measure of interest in this discussion will be the total amount of gas used. Gas is a unit measuring the computational work of running transactions or smart contracts in the Ethereum network and is a good proxy for the cost in USD. Infochain has been deployed and tested on the Ropsten Test Network, one of the commonly used public testing framework for Ethereum smart contracts. To have no limitations in terms of gas, the results reported in this paper have been generated on a local instance of Ethereum.

Dataset Description. For this experiment, we used a public dataset [Zheng *et al.*, 2014] containing real-world quality of service evaluation from 339 trusted agents for 5,825 web services. The agents observe the response time (in seconds) of the web-services. The real valued observations were placed into two categories ("good" and "bad"), in order to fit them to our binary observation setting. We treated a response time of at most 1 second as "good" and the rest as "bad". This dataset acts as the ground truth data that the information requester is interested in eliciting from self-interested agents. We simulated agent behavior as follows: 50% of the agents report truthfully, 25% report randomly (i.e. independent of the ground truth) and the rest report in an adversarial way (i.e. opposite of the truth).

Results. In Figure 2a, we show the reduction in writing cost due to the proposed optimization discussed in Section 2.2 as compared to the baseline implementation (without any proposed optimizations). Tasks in the figures refer to the questions that the agents answer. As expected, the reduction becomes more significant as agents answer more questions since the optimization can pack more and more answers into a single write operation. It may be worth noting that the optimization doesn't make writing cost independent of the number of answers as the figure may suggest. Since the number of questions in the figure doesn't exceed 42, the cost remains same as number of questions increase. Figure 2b shows the reduction in computation cost due to the proposed optimizations with varying number of agents and number of questions per agent. The figure was plotted based on the numbers obtained with the PTSC mechanism but we observed a similar trend for the OA and the DG mechanisms. We next compare the cost of the three mechanisms in Figure 2c. While the OA mechanism and the PTSC mechanism have similar cost, the DG mechanism is more costly. This is due to the fact that DG mechanism involves more operations, particularly for keeping track of questions that are not shared between agents. Finally, Figure 2d shows the effect of using randomly selected peers for reward computation in the DG mechanism. We note that there may be multiple ways to implement sampling without replacement; for e.g., 1) randomly select a peer, check if it is already in the list of previously selected peers and repeat; and 2) sample from the list of not selected peers, update the list of not selected peers and repeat. The first method is not suitable for blockchains as there is no upper bound on the number of necessary random selections and thus the transaction may run out of gas. The results presented here correspond to the second method. As shown in Figure 2d, the cost is guaranteed to reduce if we randomly select only one peer per agent. But when multiple peers are to be selected (which is required to reduce variance in rewards), the cost may increase to a level higher than the cost of using all peers without any random selection. The reason for this is that as we select more random peers, the cost of implementing random sampling exceeds the cost of simple implementation of just using all the peers.

4 Countering Lying Incentives

In the previous section, we explained how some popular peer-consistency mechanisms can be implemented in Ethereum, which is an essential step towards the design of decentralized oracles. The other crucial part is to ensure that they provide the incentives to the agents to report their observations truthfully, *even if their inherent incentives are towards the other direction*. To make this more concrete, consider the example of the the web service, mentioned in the Introduction. Clearly, the agents have an incentive to report "bad service" so that they can be compensated by the service provider. We will show how one of the peer-consistency mechanisms of Infochain, the PTSC mechanism can actually be tuned to counter these *outside* incentives. We remark that while the mechanism is known to be able to handle *constant* extra incentives [Radanovic *et al.*, 2016], like the cost of effort, it was not known whether it can be used for incentives that *depend on the outcome and the reports of the other agents*. We establish such a result in this section, and we quantify the savings that the employment of the mechanism achieves, compared to the case of not applying any peer-consistency.

Formally, we consider settings in which there is a large number of questions to be answered, and each agent selects and answers a few of them. The answer space is defined by a binary variable, e.g., "good service" or "bad service". We will use $x_i \in \{0, 1\}$ to denote the (private) observation of agent i and $y_i \in \{0, 1\}$ to denote its report. Since agents are rational, it might not be the case that $y_i = x_i$ but rather, y_i will be some function of x_i. If $y_i = x_i$ we will say that the agent is being *truthful*. Another important case is when $y_i = 0$ (regardless of x_i), where 0 denotes "bad service". The *outcome* o_q for a question q is defined as the fraction of the n agents who reported 0 as their feedback on that question. Based on the announced outcome, the agents (who submitted 0 as feedback) are issued a *refund payment* $c \cdot o_q$, i.e., proportional to the value of the outcome.

Intuitively, the outcome is determined by the agents that claimed to be dissatisfied with the service and asked for a refund. In the web service example, this corresponds to the fraction of agents who report that the response time of the service was higher than the guarantee. It should be obvious that if we do not provide any extra incentives (i.e., in the absence of peer-consistency), every rational agent would report $y_i = 0$, in order to get compensated. We will prove that with the appropriate use of PTSC, one can make sure that being truthful is the (only) best option for an agent, assuming that other agents are also truthful. In game-theoretic terms, we will prove that being truthful is a *strict equilibrium*.[3] This will be achieved via an appropriate choice of the scaling constant α in the definition of PTSC.

We have analyzed similar settings in more detail in another paper [Goel *et al.*, 2020] assuming that every agent who submit a feedback is eligible for a refund. The following analysis in this paper is for a special case when only agents who submit 0 as their feedback are eligible for a refund.

Beliefs and Belief Correlation. The most important constituents of the peer-consistency framework are the agents' beliefs about the observations of their peers. We will let $P_i(x_p = x')$, for $x' \in \{0, 1\}$, denote agent i's (prior) belief about a randomly selected peer p's observation x_p on a question being x'. A standard assumption in the literature is that the priors are fully mixed, i.e $P_i(x_p = x') > 0, \forall x' \in \{0, 1\}$. After the agent makes a private observation x_i for a question, she updates her belief (posterior) about her peer's observation on that question only, to $P_i(x_p = x'|x_i = x)$. Given the beliefs of the agents, the following quantity will be useful:

$$\beta = \min_i \Big(\frac{P_i(x_p = 1|x_i = 1)}{P_i(x_p = 1)} - \frac{P_i(x_p = 0|x_i = 1)}{P_i(x_p = 0)} \Big)$$

Intuitively, the quantity β measures the correlation strength between the observations of agents. The assumption that $\beta > 0$ is standard in the literature of peer-consistency (e.g., see [Jurca and Faltings, 2005; Witkowski and Parkes, 2012]) and in fact, it is a prerequisite for the PTSC mechanism to guarantee truthful behavior. The assumption is rather obvious

[3] Since the agents only have subjective beliefs about the observations of others, the appropriate equilibrium concept here is the *subjective equilibrium [Witkowski and Parkes, 2012]*.

in binary answer settings: if an agent observes 1, that can only increase her posterior belief about her stochastically-relevant peer also observing 1. We will make the same assumption here, and we will use β to quantify the scaling constant α that we need to use in PTSC, to overcome the lying incentives.

We will also need the following quantity:

$$\gamma = \max_i P_i(x_p = 0 | x_i = 1),$$

which measures the maximum over the posterior beliefs of any agent about her peer's observation being 0, given that her own observation was 1.

Theorem 1. *There is a value of the scaling constant α for which PTSC has a strict truthful equilibrium, even when agents have outside incentives. In particular, this is guaranteed when*

$$\alpha > \frac{c \cdot \big(1 + (n-1)\gamma\big)}{n \cdot \beta}$$

Note that the scaling constant α decreases with increasing n.

4.1 Making PTSC Profitable

While we have shown in the above theorem that agents can be incentivized to be truthful with a large-enough choice of the scaling constant of PTSC, the question that still remains is whether this is economically profitable. Are the rewards of peer-consistency so big that we end up paying more to agents, compared to what we would have paid as refund, if they all simply reported $y_i = 0$? We answer this question below. First, we quantify the payments of PTSC.

Theorem 2. *The total expected PTSC payment per agent that is enough to elicit truthful information in the presence of outside incentives is given by α.*

We note that this is not immediate from Theorem 1, as the expected payment issued for each agent is

$$\alpha \cdot \mathbb{E}\Big[\frac{\mathbb{1}_{y=y'}}{R_i(y)} - 1\Big]$$

where expectation is taken with respect to the actual random variation in the true observations of the agents (for e.g. random variation in the reception of the service). The theorem follows from the fact that this quantity in expectation can be upper bounded by 1.

Due to α's inverse dependence on n, the payments get smaller with increasing n. To provide some concrete intuition, we remark that the PTSC payment will generally only be a small fraction of the maximum outside incentive $\mathcal{R}$. For a concrete example, consider a web service that provides a good service 95% of the time; this is a realistic assumption for many web-services that stay in business. Furthermore, let us assume a small correlation in the agents' observations such that $P(x_p = 1 | x_i = 1) = P(x_i = 1) + 0.01$, and similarly for $P(x_p = 0 | x_i = 0)$. In this case, the PTSC payments per agent are just 1.2% of the refund payments c even when there are just 10 agents answering a question; the fraction quickly decreases to 0.7% of c when there are 25 agents.

In many applications, the same entity is responsible for paying the refunds as well as the PTSC payment. For example, in the web service case, the service provider is responsible for collecting truthful data and also paying the refund based on the collected payment. For such cases, we measure the relative saving of this entity as:

$$\text{relative saving: } \frac{nc - \mathcal{P}}{nc},$$

where $\mathcal{P}$ is the total payment (PTSC payments + outcome dependent payment) under the scheme to all the agents.

One might feel inclined to believe that if agents were not strategic, we could hope for a saving of 100%. However, this will be true only if the outcome is 0, when the service is "bad" 0% of the time, whereas in reality that may not be the case as we show in the next proposition. We will let $P(1)$ denote the probability that a report on the platform is 1.

Proposition 2. *If agents reported truthfully ignoring the possible refund payments, the platform could make an expected relative saving of up to $P(1)(2 - P(1))$.*

This is of course an ideal case; in the presence of rational agents, we quantify the saving of PTSC as follows.

Theorem 3. *The expected relative saving in payments made in the truth-telling equilibrium is at least*

$$P(1)(2 - P(1)) - \frac{\alpha}{c},$$

where $P(1)$ is the actual probability of a randomly selected report being 1 in the truthful equilibrium.

Note that as long as the condition $P(1)(2 - P(1)) > \alpha/c$ is satisfied, the lower bound on the saving is actually a positive number. Again, due to α's inverse dependence on n, the savings get bigger with increasing n and are also always guaranteed to be positive given sufficient value of n.

We refer the reader to [Goel *et al.*, 2020] for a discussion of the undesired equilibrium where all agents report 0, and how it can be eliminated when there exists a strictly positive fraction of honest agents. Another simple way to eliminate this undesired equilibrium is to keep open the possibility of a trusted verification if the fraction of agents reporting 0 exceeds a certain threshold. If independent verification doesn't confirm poor service, agents who reported 0 can suffer a penalty that outweighs the refunds by a large margin. A credible threat of such verification is enough to deter rational agents from playing the undesired equilibrium.

5 Conclusions

In this paper, we presented a novel system called Infochain that implements decentralized, trustelss and transparent oracles on the Ethereum blockchain. Contrary to earlier proposals on decentralized crowdsourcing systems, Infochain addresses the issue of truthfulness by implementing game-theoretic peer-consistency mechanisms. We prove that peer-consistency mechanisms can be be used to elicit truthful information even when the agents have outside incentives to misreport the information. For the first time, we discussed issues that arise in implementing these mechanisms in blockchain. The paper also presents an important new criterion for comparing or evaluating these mechanisms by their implementation complexity on the Ethereum blockchain.

References

[Adler *et al.*, 2018] John Adler, Ryan Berryhill, Andreas Veneris, Zissis Poulos, Neil Veira, and Anastasia Kastania. Astraea: A decentralized blockchain oracle. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1145–1152. IEEE, 2018.

[Agarwal *et al.*, 2017] Arpit Agarwal, Debmalya Mandal, David C. Parkes, and Nisarg Shah. Peer prediction with heterogeneous users. In *Proceedings of the 18th ACM Conference on Economics and Computation (EC)*, 2017.

[An *et al.*, 2019] Baoyi An, Mingjun Xiao, An Liu, Guoju Gao, and Hui Zhao. Truthful crowdsensed data trading based on reverse auction and blockchain. In *International Conference on Database Systems for Advanced Applications*, pages 292–309. Springer, 2019.

[Chakraborty and Das, 2016] Mithun Chakraborty and Sanmay Das. Trading on a rigged game: Outcome manipulation in prediction markets. In *IJCAI*, 2016.

[Chen *et al.*, 2011] Yiling Chen, Xi Alice Gao, Rick Goldstein, and Ian A Kash. Market manipulation with outside incentives. In *AAAI*, 2011.

[Dasgupta and Ghosh, 2013] Anirban Dasgupta and Arpita Ghosh. Crowdsourced judgement elicitation with endogenous proficiency. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013.

[Faltings and Radanovic, 2017] Boi Faltings and Goran Radanovic. Game theory for data science: Eliciting truthful information. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 11(2):1–151, 2017.

[Freeman *et al.*, 2017] Rupert Freeman, Sébastien Lahaie, and David M Pennock. Crowdsourced outcome determination in prediction markets. In *AAAI*, 2017.

[Gao *et al.*, 2016] Alice Gao, James R Wright, and Kevin Leyton-Brown. Incentivizing evaluation via limited access to ground truth: Peer-prediction makes things worse. In *2nd Workshop on Algorithmic Game Theory and Data Science at EC 2016.*, 2016.

[Goel and Faltings, 2019a] Naman Goel and Boi Faltings. Deep bayesian trust: A dominant and fair incentive mechanism for crowd. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

[Goel and Faltings, 2019b] Naman Goel and Boi Faltings. Personalized peer truth serum for eliciting multi-attribute personal data. In *35th Conference on Uncertainty in Artificial Intelligence, Tel Aviv (UAI 2019)*. AUAI, 2019.

[Goel *et al.*, 2020] Naman Goel, Aris Filos-Ratsikas, and Boi Faltings. Peer-prediction in the presence of outcome dependent lying incentives. *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*, 2020.

[Jurca and Faltings, 2005] Radu Jurca and Boi Faltings. Enforcing truthful strategies in incentive compatible reputation mechanisms. In *International Workshop on Internet and Network Economics*, pages 268–277. Springer, 2005.

[Kong *et al.*, 2019] Yuqing Kong, Yiping Ma, and Yifan Wu. Securely trading unverifiable information without trust. *arXiv preprint arXiv:1903.07379*, 2019.

[Li *et al.*, 2018] Ming Li, Jian Weng, Anjia Yang, Wei Lu, Yue Zhang, Lin Hou, Jia-Nan Liu, Yang Xiang, and Robert H Deng. Crowdbc: A blockchain-based decentralized framework for crowdsourcing. *IEEE Transactions on Parallel and Distributed Systems*, 30(6):1251–1266, 2018.

[Liu and Chen, 2017] Yang Liu and Yiling Chen. Machine-learning aided peer prediction. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 63–80. ACM, 2017.

[Lu *et al.*, 2018] Yuan Lu, Qiang Tang, and Guiling Wang. Zebralancer: Private and anonymous crowdsourcing system atop open blockchain. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 853–865. IEEE, 2018.

[Miller *et al.*, 2005] Nolan Miller, Paul Resnick, and Richard Zeckhauser. Eliciting informative feedback: The peer-prediction method. *Management Science*, 51(9):1359–1373, 2005.

[Prelec, 2004] Dražen Prelec. A bayesian truth serum for subjective data. *Science*, 306(5695):462–466, 2004.

[Radanovic *et al.*, 2016] Goran Radanovic, Boi Faltings, and Radu Jurca. Incentives for effort in crowdsourcing using the peer truth serum. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 7(4):48, 2016.

[Ryan, 2017] Danny Ryan. Calculating costs in ethereum contracts, 2017.

[Shnayder *et al.*, 2016] Victor Shnayder, Arpit Agarwal, Rafael Frongillo, and David C Parkes. Informed truthfulness in multi-task peer prediction. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 179–196. ACM, 2016.

[Waggoner and Chen, 2014] Bo Waggoner and Yiling Chen. Output agreement mechanisms and common knowledge. In *Second AAAI Conference on Human Computation and Crowdsourcing*, 2014.

[Witkowski and Parkes, 2012] Jens Witkowski and David C Parkes. Peer prediction without a common prior. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 964–981. ACM, 2012.

[Xiong and Xiong, 2019] Wei Xiong and Li Xiong. Smart contract based data trading mode using blockchain and machine learning. *IEEE Access*, 2019.

[Zheng *et al.*, 2014] Zibin Zheng, Yilei Zhang, and Michael R Lyu. Investigating qos of real-world web services. *IEEE transactions on services computing*, 7(1):32–39, 2014.