Improving the Resiliency of Decentralized Crowdsourced Blockchain Oracles

Adrian Fuertes Blanco[1], Zeshun Shi[1,2][0000−0001−9163−8023], Debraj Roy[1][0000−0003−1963−0056], and Zhiming Zhao[1][0000−0002−6717−9418]

[1] Informatics Institute, University of Amsterdam, Amsterdam, the Netherlands
adrifuertes@me.com, d.roy@uva.nl, z.zhao@uva.nl
[2] Cyber Security Group, Delft University of Technology, the Netherlands
z.shi-2@tudelft.nl

Abstract. The emergence of blockchain technologies has created the possibility of transforming business processes in the form of immutable agreements called smart contracts. Smart contracts suffer from a major limitation; they cannot authenticate the trustworthiness of real-world data sources, creating the need for intermediaries called oracles. Oracles are trusted entities that connect on-chain systems with off-chain data, allowing smart contracts to operate on real-world inputs in a trustworthy manner. A popular oracle protocol is a crowdsourced oracle, where unrelated individuals attest to facts through voting mechanisms in smart contracts. Crowdsourced oracles have unique challenges: the trustworthiness and correctness of outcomes cannot be explicitly verified. These problems are aggravated by inherent vulnerabilities to attacks, such as Sybil attacks. To address this weakness, this paper proposes a reputation-based mechanism, where oracles are given a reputation value depending on the implied correctness of their actions over time. This reputation score is used to eliminate malicious agents from the participant pool. Additionally, two reputation-based voting mechanisms are proposed. The effectiveness of the proposed mechanism is evaluated using an agent-based simulation of a crowdsourced oracle platform, where a pool of oracles performs evaluate Boolean queries.

Keywords: Blockchain · Reputation-based consensus · Decentralized oracle · Voting.

1 Introduction

Blockchain technologies can enhance business processes by overcoming the requirement of trust in contracts through the use of smart contracts. Smart contracts are programs stored on the blockchain; each invocation that modifies the state of the smart contract will be recorded on the blockchain, making the transaction immutable and visible to all authorized parties [4]. While smart contracts can bring trustworthiness and transparency to business processes, they have one major limitation: they cannot directly authenticate external APIs and make use

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

of real-world data. Trusted entities, called oracles, act as an interface between the real-world and smart contracts [12].

There exist different types of oracles, depending on the process at hand. Software oracles can extract data from applications (i.e. web services), hardware oracles can extract data from the physical world (i.e. IoT devices) and crowdsourced oracles can leverage the power of human judgment to make decisions (i.e. data labeling) [6]. Oracles usually utilize multiple independent data sources, and the source quality and integrity is a core cause of concern in smart contracts [2]. To make the process sustainable and to incentivize participation, oracles usually charges a fee for each transaction. The remuneration of oracle tasks can then be modeled as an economic problem in a market, with oracles exchanging their labor for a fee. While the market structure creates an incentive for efficient participation, it also creates an incentive for malicious interference. The potential for malicious interference depends on the protocol used and the oracle type. In this paper, we considered the most popular permissionless, voting-based crowdsourced oracles that operate binary tasks (queries with a True or False answer).

The crowdsourcing process involves requesters posting tasks to be completed by workers. Tasks can be unrelated and workers can choose or be assigned tasks. In decentralized oracles, requesters and workers are anonymous and independent. To ensure correctness, multiple workers are assigned to a task and the majority vote is selected as the final answer. Workers who vote with the majority are rewarded, while others who don't might lose their deposit. Voting systems can improve truth-telling and oracle accuracy but create vulnerabilities, such as manipulation of majority results by malicious actors [16]. These vulnerabilities can be addressed through incentive structures that make malicious behavior unprofitable or through trust management by estimating the trustworthiness of individuals and designing voting and participation rules accordingly.

In this paper, three reputation-based mechanisms are introduced to improve the resiliency of crowdsourced oracles by controlling the participation of agents and altering the voting mechanism based on reputation. A reputation system is first designed to reward participants who vote truthfully and eliminate participants who act maliciously. The simple majority voting system is replaced by a reputation-based weighted voting system, which is designed to limit the influence of malicious agents in the event of attacks. Two variations of this system are further proposed to tackle Sybil and Camouflage Attacks, respectively. Additionally, an agent-based simulation of a crowdsourced oracle platform is introduced. The simulation is used to validate the aforementioned strategies, providing insight into the resiliency of the system under various attacks. All of the reputation-based mechanisms are tested under Simple Attacks, Camouflage Attacks, and Sybil Attacks. Lastly, a novel "certifier" agent is defined as a bounded-rational, profit-maximizing player, which utilizes past voting outcomes and innate accuracy in a probabilistic decision-making model. This design improves upon existing methods by relaxing rationality assumptions and incorporating stochasticity into the decision-making process, characteristic of crowdsourcing systems. This agent is used in the agent-based simulation in an honest certifier role.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

The rest of the paper is organized as follows: Section 2 discusses related work on the resilience of crowdsourced oracles; Section 3 presents the reputation-based model and details the simulation components; Section 4 displays the simulation settings and experiment results; Section 5 discusses the obtained results and Section 6 concludes the paper.

2 Related Work

The existing frameworks for decentralized crowdsourced oracles can be divided on whether they tackle vulnerabilities through incentive design or reputation management. Frameworks that focus on incentive design, such as [1], [12] and [15], propose guidelines on reward quantities [15] or variations on the majority voting system [1] [12] designed to make malicious behavior unprofitable for rogue agents. These frameworks argue their incentive systems induce truth-telling by proving honesty is optimal in a Nash Equilibrium. The optimality proof requires the assumption that all participants are fully rational and have a sufficiently large accuracy, which might not hold for all agents or tasks in reality [3].

These vulnerabilities still leave room for a reputation-based system to control unwanted behavior. To our best knowledge, no explicit reputation-based control system has been proposed for permissionless crowdsourced oracles. Though, decentralized crowdsourcing platforms [10] [9] outline reputation systems to optimize worker [10] and requester [9] behavior, their systems (like traditional crowdsourcing) require the verification of output by a third party. Output verification is useful in a crowdsourcing context, where output quality is easily verifiable, but it is not feasible to implement in oracle models, where the underlying oracle output is the closest thing to ground truth itself. This creates the need for a reputation-based system that is not based on external verification and can be supported by the oracle's output.

Reputation-based voting is a popular form of achieving consensus in governance systems such as [13]. Reputation is used both as a way to deter unwanted behavior and a way to reward active participants. As users gain relative reputation, their influence increases, since reputation acts as weight in a weighted voting system. While [15], [12] and [1] suggest the use of a reputation-based system might be a beneficial addition to their incentive systems, no explicit implementation details are suggested.

A major shortcoming in all the aforementioned protocols is the lack of practical evidence. While incentive optimization techniques might suggest optimal parameters, experimental evidence suggests human behavior might not align with theoretical predictions [3]. The implementation of blockchain-based systems can be very costly [8], a problem accentuated by the immutability of smart contracts. This immutability is particularly problematic in incentive design and reputation management, where parameter choice can have a significant impact on system performance. This results in a higher potential cost of failure, on top of the high implementation cost.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

A solution to high smart contract development costs is using simulation systems like Agent-Based Model (ABM). They study the properties of complex systems and have been used for crowdsourcing, voting, and smart contract manipulation. Simulation allows developers to test parameters, strategies, and threats before deployment, reducing cost and uncertainty [7].

3 Proposed Model

This section explains the components of the proposed model, starting with a system overview. The design of the agents is covered afterward, followed by a description of reputation voting and control models. Lastly, experiment evaluation metrics are discussed.

3.1 System Overview

There are two roles in a decentralized crowdsourced oracle: requesters and certifiers. Within the oracle platform, requesters submit queries (q) with an associated budget to them. Certifiers can then choose to engage in these queries to potentially earn a reward. We assume each query q has an underlying true value of True (T) or False (F). The objective of the oracle is to reach this true value via a voting mechanism performed by the certifiers. Since the true value is not known, certifiers are rewarded based on the value estimated by the oracle. To incentivize honesty, certifiers submit a participation fee that they are liable to lose if their behavior does not align with the oracle's output. The oracles considered in this paper are permissionless, meaning no approval is required to join the system. To join the pool, certifiers do have to register with their wallet address, which gives them a starting reputation and the ability to participate in games.

We assume the queries are of a monitoring nature, meaning a query is constantly being evaluated by requesters and it's bound to change at any moment. As noted by [14,15], an example of this would be monitoring the compliance of a Service Level Agreement (SLA) for cloud computing services, where the oracle is used to attest for the up-time of the service (certifying whether a violation in the SLA has occurred or not). These tasks are characterized by a significantly higher likelihood of the true value being $False$ (no violation has occurred), meaning true $Trues$ are rare and not guaranteed. For this reason, it is assumed that the reward associated with voting $False$ is lower than the reward associated with voting $True$, as agents are then incentivized to exert more effort when a $True$ outcome is observed, avoiding a lazy equilibrium in which every agent reports $False$ for a guaranteed income with minimum effort.

In this context, malicious agents are incentivized to influence the oracle towards a $False$ outcome, since it is likely to go unnoticed and it guarantees a low but steady reward. Adversarial agents can influence requesters by either consistently or overwhelmingly reporting $False$. If these attacks are successful, honest certifiers (who are assumed to be profit-maximizing agents) will be incentivized to also submit $False$ reports, as the expected profit of doing so would be higher

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

than that of true reporting. We consider three different threat models which will be described in detail in Section 3.4. Besides, the reputation mechanism is designed to limit malicious agent influence in two ways: by excluding agents with a bad reputation and by limiting their voting power. The iteration of the voting game is shown in Figure 1, and the implementation details are discussed in Section 3.3.

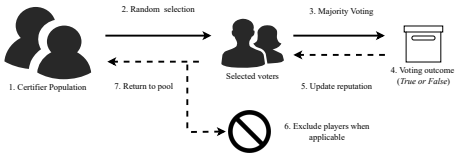

Fig. 1. Diagram showcasing a single iteration of the voting game.

3.2 Agents

Certifiers. Certifiers are assumed to be honest and profit-maximizing players. Each player i is assumed to have an innate accuracy value $\gamma_i \in [0, 1]$. Intuitively this value represents the ability of the player to assess the query at hand, and therefore the probability they will be correct about their assessment. Players use their accuracy to form a private assessment of the query. Additionally, each player has a fixed memory value m_i ($m_i \geq 0$). Their memory represents their ability to look at past votes, either through the public transactions stored on the blockchain or through the memory of past games. An overview of the decision-making process behind agent's votes is outlined in Figure 2.

Each agent balances both their innate accuracy and their memory of past votes to create a profit expectation of voting truthfully (submitting their private opinion) or not (submitting the opposite of their private opinion). The balance between these two parameters is parametrized by a recency bias $RB \in [0, 1]$, which acts as a weight between the profit estimate from private opinion and the profit estimate from the available voting history. Formally, a certifier A_i with accuracy γ_i, memory m_i and recency bias RB_i submits a vote $v_i \in [F, T]$. A_i will submit T if $E[\pi(v_i = T)] \geqslant E[\pi(v_i = F)]$, otherwise they will submit F.

The expected profit of submitting T is calculated as:

$$E[\pi(v_i = T)] =$$
$$Pr(MV = T \mid v_i = T) \cdot \pi(MV_{v_i = MV} = T) +$$
$$(1 - Pr(MV = T \mid v_i = T)) \cdot \pi(MV_{v_i \neq MV} = F)$$

where $MV \in [T, F]$ represents the majority vote, or the vote chosen by the majority voting process.

If $v_i = T$, the probability of a consensus being formed is:

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

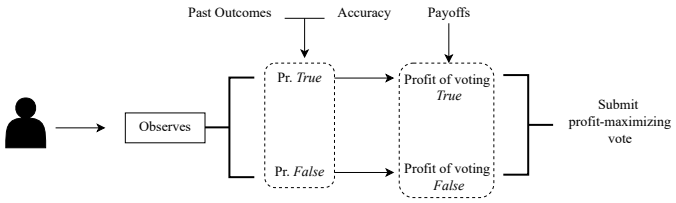

Fig. 2. Overview of the decision-making process of an honest certifier.

$$Pr(MV = T \mid v_i = t) =$$
$$RB_i \cdot Pr(MV = r \mid a_i = r, memory = m_i) + (1 - RB_i) \cdot \gamma_i$$

Where $Pr(MV = FT \mid v_i = T, memory = m_i)$ measures the expected likelihood of the consensus being formed around F, given the votes available to the players' memory. This is modeled using a Beta-Binomial distribution as such:

$$Pr(MV = T \mid v_i = T, memory = m_i)$$
$$= 1 - BinomialCDF(p, k, n)$$

The p parameter follows a $Beta(\alpha, \beta)$ with $\alpha = \sum_{i=1}^{n_{memory}} 1 \cdot [v_i = True]$ and $\beta = \sum_{i=1}^{n_{memory}} 1 \cdot [v_i = False]$. Then, $k = n_{players} \cdot MajorityThreshold - 1$, with the majority threshold being the percentage of votes required for consensus in a game (50% for the simple majority). Lastly, n is the number of players in a game.

The profit estimation is identical for $E[\pi(v_i = F)]$, though the payoffs change and the accuracy value is reversed since it is assumed to refer to the accuracy of assessing a true $True$ signal.

For the purpose of the simulation, we assume all agents share the same memory and recency bias. These values are set at 10 for memory, such that agents have access to the votes cast in the last 10 games, and 0.3 for the recency bias, such that agents weigh their experience 30% while making a profit estimation. The accuracy is sampled from a Beta distribution with $\alpha = 10$ and $\beta = 2.75$, which creates a visual center around 0.85 accuracy, as observed from Figure 3. This ensures the majority of agents have relatively high accuracy, with some exceptions. Lastly, we assume a simple majority is necessary to achieve consensus.

This formulation ensures that both an agent's accuracy as well as their experience are considered in the decision-making process. Additionally, the modeling of accuracy creates more realistic agents, as real participants are not always 100% accurate and are liable to make mistakes.

Malicious. Malicious agents are assumed to have 100% accuracy and they always follow their assigned voting strategy regardless of expected profits. The voting strategies of malicious agents are outlined in Section 3.4.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

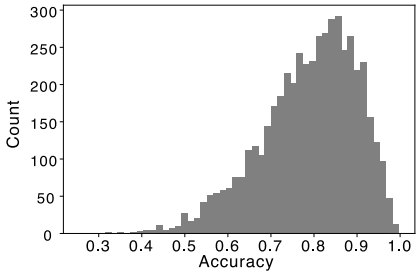

Fig. 3. Histogram including 5000 sampled values from a $Beta(\alpha, \beta)$ distribution with $alpha = 10$ and $beta = 2.75$

3.3 Reputation

For each agent, their reputation value $R_i, R_i \geq 0$ represents their trustworthiness based on past voting outcomes. All certifiers start with a reputation value $R_s, R_s > 0$. As players participate in games, their reputation can increase by R_+ if they vote according to the consensus, or decrease by R_- if they vote against the consensus. R_s, R_+ and R_- are model parameters.

Participation Control. As a way to control malicious agents, agents will be permanently excluded from the system if their reputation value reaches zero. This can prevent agents from engaging in a Simple Attack, where agents continuously perform adversarial behavior. Since the behavior is constant and individual, the attacker is likely to vote against the consensus enough times for them to be expelled from the pool. This holds true as long as the percentage of the malicious agent is less than 50%, such that the oracle is likely to produce the correct result on average.

This does not prevent agents from engaging in Camouflage Attacks or Sybil Attacks. Agents involved in Camouflage Attacks pretend to be honest until they achieve a high reputation status when they start attacking the system. Given their high reputation, if enough malicious agents are in the system, it's possible they will influence the oracle's opinion before their reputation goes to zero. In Sybil Attacks, an attacker impersonates a large number of players in order to influence game outcomes. If the attacker floods the pool with enough agents, such that over 51% of the players in a game are malicious, the oracle will always vote in accordance with the malicious agents, making it such that their reputation never reaches zero.

Voting Systems. Reputation-based, weighted voting systems can mitigate Camouflage and Sybil Attacks by redistributing voting power and minimizing the voting influence of malicious agents. In a simple voting system, every vote has the same weight, which results in every voter having the same voting power.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

In a system with 10 voters, gaining control over 4 votes can already give an attacker decision power over the system outcome. By weighting votes by a user's reputation value, higher voting power is assigned to voters with a proven history of truthful behavior. This mitigates Sybil Attacks, since all newly created player accounts will have a relatively low reputation, assuming more experienced players are participating in the game. This would increase the necessary amount of players an attacker has to control, which given the random player selection and a sufficiently large user base, would make this attacks impossible.

The weight of each candidates vote, x_i, is calculated as follows:

$$x_i = \frac{R_i}{\sum_j^n R_j}, i \neq j$$

The total votes for each outcome can be calculated as follows:

$$V_{True} = \sum_{i=1}^{n} x_i \cdot v_i[v_i = True]$$

$$V_{False} = \sum_{i=1}^{n} x_i \cdot v_i[v_i = False]$$

Then, if $V_{True} \geq V_{False}$ the outcome will be decided as $True$. Otherwise, the outcome will be decided as $False$.

While this offers an improvement upon simple voting, weighted voting suffers from seniority-induced advantages which can disincentive participation. This is known as the Matthew Effect, which dictates that those who begin with advantage accumulate more advantage over time and those who begin with disadvantage become more disadvantaged over time [5]. The earlier an honest certifier joins the platform, the more opportunities they will have to gain a reputation. This leads to the earliest certifiers having disproportionate voting power over newcomers, which also increases the potential negative influence they can have over the system if they act maliciously. For example, a player who has played 1010 games, can have 1000 times more voting power than a certifier who has played 10 games. Despite players not having access to others' reputation scores, this could easily disincentive new players from joining the certifier pool. Stratified voting can solve Matthew Effect by weighting votes relative to the game participants, rather than to the whole certifier pool. This can be achieved by dividing the pool into k different partitions, where agents in each partition are given the same voting weight. Agents are assigned to partitions depending on their global reputation value, but since the divisions are made relative to the participants in the game, an agent with extensive experience can't have disproportionate voting power, since their voting power will be the same as the second most reputed player in the game. The number of partitions k is a model parameter and it can be set depending on the number of participants. The relative vote weight assigned to each partition, w_k, is also a model parameter. The combination of k and w_k can affect the ultimate voting power of each agent, though finding the

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

optimal combination is out of the scope of this paper. For the purpose of this simulation, a linear weight scale will be tested.

To assign agents to a partition, participants are sorted based on their reputation score and assigned to partition 1 to k, or the closest possible number. The number of partitions, k, acts as an upper bound, since it cannot be guaranteed that agents will have different reputation values with random sortition.

Each agent's weight would then be calculated as follows:

$$x_i = w_k, A_i \in k$$

$$w_k = \sigma(k)$$

Where the w_k is the weight of the partition A_i belongs to and σ defines the weight scale depending on the partition number. For a linear partition $w_k = k$.

3.4 Threat Models

Three different threat models are considered in this paper: Simple Attacks, Camouflage Attacks, and Sybil Attacks. Each reputation-based control mechanism will be assessed under all three threat models.

Simple Attacks. In a Simple Attack, a malicious agent exists passively in the agent pool, voting $False$ whenever it enters a game. In this attack, a percentage of the agents in the pool, $p_a \in [0, 1]$, are malicious, and as $p_a \to 0.5$ their influence surpasses that of honest agents, inducing the oracle to always output $False$.

Camouflage Attacks. In the Camouflage Attack, a malicious agent masquerades as an honest agent for n_c turns or until it has accrued a high enough reputation value, R_t. When $R_i \geq R_t$ the agent starts attacking the system by always voting $False$. Since malicious agents are assumed to be 100% accurate $n_c = R_t$ in this model. n_c and R_t are model parameters.

Sybil Attacks. In a Sybil Attack, an attacker forges multiple accounts, flooding the agent pool with malicious agents that always vote $False$. The number of malicious agents that enter the pool is parameterized as $n_f = \alpha n$, where n is the total number of agents in the pool. Since in this model game participants are randomly selected, α should be larger than one (under simple voting) to perform a successful attack. The attack lasts for $i_a \in [1, \text{inf}]$ turns.

3.5 Rewards

Since the game is assumed to be monitoring in nature, the reward for a $True$ outcome, π_{TT}, is m times larger than that of a $False$ outcome, π_{FF}. We follow

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

the payment structure outlined in [15] $\pi_{FF} = 1$, and an agent loses the full amount of their deposit if they vote $False$ when the oracle outputs $True$, making $\pi_{FT} = -1$. An agent faces no loss when the oracle outputs $True$ and they vote $False$, making $\pi_{TF} = 0$. We set $m = 2$, instead of the proposed $m = 10$, which increases the influence of attacks while keeping the system accurate without threats. The payoff matrix is outlined in Table 1.

Table 1. Payoff matrix relating an agent's payoff relative to the oracle output

		Consensus	
		$True$	$False$
Player i	$True$	2	-1
	$False$	0	1

3.6 Evaluation

Ground Truth Model. To model accuracy, the underlying truth distribution needs to be defined. The underlying truth is modeled as a sign square wave, which oscillates between $True$ (-1) and $False$ (1) in phases. The formulation is as follows:

$$S(t) = \text{sgn}(\sin 2\pi f t)$$

With the square wave following a duty cycle as $duty = \frac{sin(m\pi t+1)}{2}$. m has been fixed to 38.8 and f has been fixed to 58 with $t = 500$. This results in an average of 40% $True$ periods, such that the majority of agents are able to experience both outcomes, while the True outcome remains less frequent. A sample of this signal for 500 periods is shown in Figure 4.

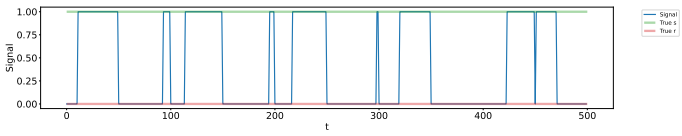

Fig. 4. Model of a ground truth signal using a square sign wave over 500 periods. The signal oscillates between 1 and -1, indicating a $False$ and $True$ actual signal.

Accuracy. The primary measure of success for a single experiment in a simulation is its accuracy in the final period. The accuracy is measured as the percentage of iterations in which the oracle predicted the ground truth. Formally:

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

$$accuracy = \frac{\sum_{i=1}^{n} 1 \cdot [o_i = t_i]}{n}$$

Where n refers to the total number of iterations, o_i refers to the oracles output in iteration i and t_i refers to the true signal in iteration i.

4 Experiments and Simulation

This section outlines the details of each experiment performed in this paper followed by the results.

4.1 Simulation Settings

All simulation experiments share the following parameters:

- 15 voters are randomly selected for each game.
- Each player follows the payoffs aligned in Section 3.5.
- Each simulation follows the true signal outlined in Section 3.6.
- The voting power distribution is calculated over the last 5 games played by each agent.

The number of game participants is chosen to be odd which prevents ties during voting. Additionally, it is sufficiently large such that an average player will experience enough games to gain a meaningful reputation (or be banned) and have ample voting opportunities, which could influence their voting choices.

Each of the proposed mechanisms is tested using a Simple Attack, a Camouflage Attack, and a Sybil Attack; all compared against a baseline with no interventions. The Simple Attack is simulated with $p_a \in [0.05, 0.50]$ in 0.025 intervals. The Camouflage Attack is simulated with $n_c = 5$ and $p_a \in [0.05, 0.50]$ in 0.025 intervals. Lastly, the Sybil Attack is simulated with $\alpha = 3$ and $i_a \in [0, 65]$ in 5 game intervals. Each Sybil Attack simulation includes 5 attacks. The timing of attacks is random, with at least 50 games between attacks, such that the system is under attack at most 50% of the time.

Each attack is tested with increasing p_a (Simple and Camouflage) or i_a (Sybil) to test the resilience of the system, as an increased presence of malicious agents impedes the oracle from achieving high accuracy.

Each agent i starts with 1 reputation point, $R_s = 1$, with $R_+ = 1$ and $R_- = 1$. On average, an agent can accumulate 15 reputation points, and they can last a minimum of 1 turn in the agent pool, under participation control.

Due to the stochastic nature of agent decision-making and agent accuracy sampling, each experiment consists of multiple iterations, with the results being presented as a sample average. Specifically, each experiment consists of 100 iterations, with 500 consecutive games per iteration. In each iteration, a pool of 500 players is generated.

The simulations are built in Python, using the ABM framework MESA [11].

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

4.2 Participation Control

In the participation control experiments, agents are permanently banned from the agent pool when their reputation reaches 0, with the intention of spelling malicious agents when they vote maliciously. The experiment results for each attack tested are displayed in Figure 5.

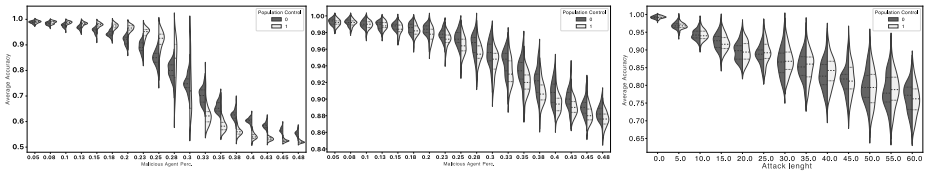

Fig. 5. Participation control simulation results with subfigures displaying the Simple, Camouflage, and Sybil attacks respectively. The violin plots show the density of average accuracy values from control and treatment simulations, each with 100 observations.

Simple Attack. The percentage of malicious agents can significantly affect the accuracy of the system under a Simple Attack, with average accuracy values decreasing to 50% as p_a approaches 0.5. The participation control system helps mitigate this decline under lower threat levels, resulting in higher average accuracy up to $p_a \sim 0.25$. When $p_a \in [0.25, 0.3]$, the accuracy under the treatment becomes uniformly distributed - meaning sampled accuracy is as often very high as it is very low. In the remaining p_a percentages, the population-controlled simulations have significantly lower accuracy than the control. **Camouflage Attack.** The reputation control strategy does not improve resiliency when compared to the control, with the average treatment accuracy being lower than the control. The lower accuracy is caused by mistake-making honest being eliminated, which then increases the relative percentage of malicious agents once the camouflage period is over. **Sybil Attack.** The reputation control mechanism does not improve resiliency against the simulated Sybil Attacks.

4.3 Weighted Voting

In the weighted voting experiments, simple majority voting is replaced with weighted voting, with reputation values acting as weights. By giving more reputable agents (which are likely to be honest certifiers) higher voting power, the oracle is expected to produce more accurate results under threat. The experiment results for each attack tested are shown in Figure 6.

Simple Attack. The weighted voting strategy increases resiliency compared to the control, being able to maintain a higher average accuracy under larger p_a values. The strategy results in significantly increased accuracy until $p_a \sim 0.35$, and maintains a relatively higher accuracy for the remaining p_a. **Camouflage**

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

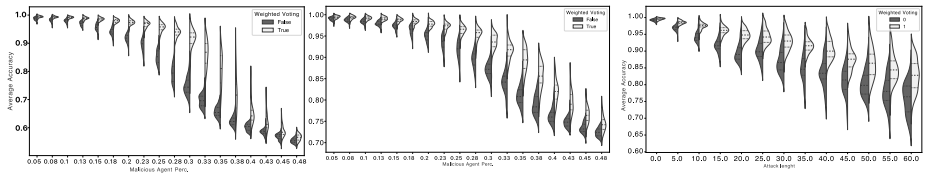

Fig. 6. Weighted voting simulation results with subfigures displaying the Simple, Camouflage, and Sybil attacks respectively. The violin plots show the density of average accuracy values from control and treatment simulations, each with 100 observations.

Attack. The weighted voting strategy consistently achieves higher average accuracy than the control, demonstrating it is effective in increasing resiliency against a Camouflage Attack. **Sybil Attack.** A weighted voting system can improve resiliency against Sybil Attacks. Although simulations using this system show increased average accuracy under all attack lengths, the differences are lower than under other attacks.

4.4 Stratified Voting

In the stratified voting experiment, agents are assigned to k partitions, which dictate their relative voting power. Each game will consist of up to 5 partitions ($k = 5$) with linear voting power differences between partitions. By assigning linear weights between partitions, individual agents have less voting power when compared to weighted voting, which is intended to limit the voting power of more experienced agents. The experiment results for each attack tested are shown in Figure 7.

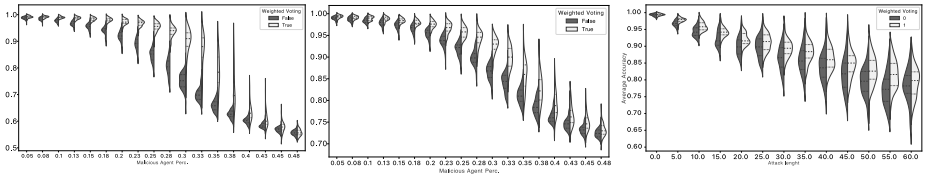

Fig. 7. Stratified voting simulation results with subfigures displaying the Simple, Camouflage, and Sybil attacks respectively. The violin plots show the density of average accuracy values from control and treatment simulations, each with 100 observations

Simple Attack. The stratified voting strategy increases resiliency when compared to the control. Simulations using stratified voting result in significantly higher accuracy until $p_a \sim 0.3$, maintaining an average accuracy higher than 90%. From $p_a \sim 0.3$, the stratified voting simulations result in higher or equal

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

average accuracy. **Camouflage Attack.** The stratified voting strategy shows higher average accuracy than the control under a Camouflage Attack. **Sybil Attack.** The Stratified voting strategy can increase the oracles' resiliency under a Sybil Attack. The efficacy is lower than that showcased by the weighted voting strategy, particularly for higher attack lengths.

5 Discussion

Results from all experiments show that an increase in malicious agents leads to a decline in oracle accuracy across all threat models, revealing the vulnerability of the crowdsourced oracle without threat prevention. Participation control improves resiliency for lower p_a levels under Simple Attack, however it worsens accuracy under Camouflage and Simple Attacks due to malicious agents exploiting the control mechanism. Alternative voting mechanisms improve resiliency against Simple Attacks, but still produce low accuracy as p_a approaches 0.5. Simple weighted voting is the only mechanism to significantly improve resiliency under Sybil Attacks, due to the importance of players with large reputation values. The high accuracy seen under Camouflage Attack suggests that 500 game periods are insufficient to observe Matthew Effect. The efficacy of Weighted and stratified voting systems remains to be tested over longer simulation cycles.

6 Conclusion

In this paper, we propose three reputation-based methods to improve the resiliency of crowdsourced oracles. We leverage simulation techniques to test the validity of the proposed methods under multiple threat models. A bounded-rational agent model was also introduced, enabling the simulation to capture oracles dynamics over time. Finally, we discuss the strengths and weaknesses of the mechanism proposed, highlighting the mechanism behind them based on the simulation results.

In future work, we plan to expand the simulation scope by modeling the self-selection process of participants to individual games. This is expected to affect all reputation-based mechanism, as agents are incentivized to explicitly increase their reputation scores in order to gain voting power (which is not possible under random selection). The direction of this effect (increasing or decreasing resilience) is expected to depend on the integrity of the agent pool, as self-selection will empower honest agents and dishonest agents alike, possibly leading to extreme results in either direction. We also aim to implement different weighting functions for the stratified voting method, as results suggest that relative voting power differences are a key aspect of the improvement in resilience. Lastly, we plan to conduct empirical experiments on crowdsourced oracle systems. During this experiment, we aim to identify the voting and selection pattern of certifiers, which would serve to inform the selection and decision-making process of the model. These experiments would also serve as validation of the simulation model.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1

Acknowledgements This research is funded by the European Union's Horizon 2020 research and innovation program under grant agreements 825134 (ARTICONF project), 862409 (BlueCloud project) and 824068 (ENVRI-FAIR project). The work is also partially supported by LifeWatch ERIC.

References

1. Adler, J., Berryhill, R., Veneris, A., Poulos, Z., Veira, N., Kastania, A.: Astraea: A decentralized blockchain oracle. arXiv:1808.00528 [cs] (Aug 2018), arXiv:1808.00528
2. Caldarelli, G., Ellul, J.: The blockchain oracle problem in decentralized finance—a multivocal approach. Applied Sciences **11**(1616), 7572 (Jan 2021)
3. Camerer, C.F.: Behavioral game theory: Experiments in strategic interaction. Princeton university press (Mar 2003)
4. Chen, Y., Meng, L., Zhou, H., Xue, G.: A blockchain-based medical data sharing mechanism with attribute-based access control and privacy protection. Wireless Communications and Mobile Computing **2021**, 1–12 (2021)
5. Dannefer, D.: Cumulative advantage/disadvantage and the life course: Cross-fertilizing age and social science theory. The Journals of Gerontology: Series B **58**(6), S327–S337 (Nov 2003)
6. Di Ciccio, C., Meroni, G., Plebani, P.: On the adoption of blockchain for business process monitoring. Software and Systems Modeling **21**(3), 915–937 (Jun 2022)
7. Janssen, M.A., Ostrom, E.: Empirically based, agent-based models. Ecology and Society **11**(2) (2006)
8. Khan, S.N., Loukil, F., Ghedira-Guegan, C., Benkhelifa, E., Bani-Hani, A.: Blockchain smart contracts: Applications, challenges, and future trends. Peer-to-Peer Networking and Applications **14**(5), 2901–2925 (Sep 2021)
9. Li, C., Qu, X., Guo, Y.: Tfcrowd: a blockchain-based crowdsourcing framework with enhanced trustworthiness and fairness. EURASIP Journal on Wireless Communications and Networking **2021**(1), 168 (Aug 2021)
10. Li, M., Weng, J., Yang, A., Lu, W., Zhang, Y., Hou, L., Liu, J.N., Xiang, Y., Deng, R.H.: Crowdbc: A blockchain-based decentralized framework for crowdsourcing. IEEE transactions on parallel and distributed systems **30**(6), 1251–1266 (2019)
11. Masad, D., Kazil, J.: Mesa: An agent-based modeling framework. In: 14th PYTHON in Science Conference. p. 51–58 (Jan 2015)
12. Nelaturu, K., Adler, J., Merlini, M., Berryhill, R., Veira, N., Poulos, Z., Veneris, A.: On public crowdsource-based mechanisms for a decentralized blockchain oracle. IEEE Transactions on Engineering Management **67**(4), 1444–1458 (Nov 2020)
13. Orange: https://www.orangeprotocol.io/
14. Shi, Z., Farshidi, S., Zhou, H., Zhao, Z.: An auction and witness enhanced trustworthy sla model for decentralized cloud marketplaces. In: Proceedings of the Conference on Information Technology for Social Good. pp. 109–114 (2021)
15. Zhou, H., Ouyang, X., Ren, Z., Su, J., de Laat, C., Zhao, Z.: A blockchain based witness model for trustworthy cloud service level agreement enforcement. In: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications. p. 1567–1575 (Apr 2019)
16. Zhuang, Q., Liu, Y., Chen, L., Ai, Z.: Proof of reputation: A reputation-based consensus protocol for blockchain based systems. In: Proceedings of the 2019 International Electronics Communication Conference. p. 131–138. ACM, Okinawa Japan (Jul 2019)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_1