Multi-round Master-Worker Computing:
a Repeated Game Approach

Antonio Fernández Anta
IMDEA Networks Institute,
Madrid, Spain,
antonio.fernandez@imdea.org

Chryssis Georgiou
University of Cyprus,
Nicosia, Cyprus,
chryssis@cs.ucy.ac.cy

Miguel A. Mosteiro, Daniel Pareja
Kean University,
Union, NJ, USA,
{mmosteir,parejad}@kean.edu

Abstract—We consider a computing system where a master processor assigns tasks for execution to worker processors through the Internet. We model the workers' decision of whether to comply (compute the task) or not (return a bogus result to save the computation cost) as a mixed extension of a strategic game among workers. That is, we assume that workers are rational in a game-theoretic sense, and that they randomize their strategic choice. Workers are assigned multiple tasks in subsequent rounds. We model the system as an infinitely repeated game of the mixed extension of the strategic game. In each round, the master decides stochastically whether to accept the answer of the majority or verify the answers received, at some cost. Incentives and/or penalties are applied to workers accordingly.

Under the above framework, we study the conditions in which the master can reliably obtain tasks results, exploiting that the repeated game model captures the effect of long-term interaction. That is, workers take into account that their behavior in one computation will have an effect on the behavior of other workers in the future. Indeed, should a worker be found to deviate from some agreed strategic choice, the remaining workers would change their own strategy to penalize the deviator. Hence, being rational, workers do not deviate.

We identify analytically the parameter conditions to induce a desired worker behavior, and we evaluate experimentally the mechanisms derived from such conditions. We also compare the performance of our mechanisms with a previously known multi-round mechanism based on reinforcement learning.

Keywords-Internet Computing; Master-Worker Task Computing; Game Theory; Repeated Games; Algorithmic Mechanism Design;

I. INTRODUCTION

Motivation and prior work. The processing power of top supercomputers has reached speeds in the order of PetaFLOPs [1]. However, the high cost of building and maintaining such multiprocessor machines makes them accessible only to large companies and institutions. Given the drastic increase of the demand for high performance computing, Internet-based computing has emerged as a cost-effective alternative. One could categorize Internet-based computing into two categories: *administrative* computing and *master-worker* computing. In the first one, the computing elements are under the control of an administrator. Users of such infrastructure have to bear the cost of access, which depends on the quality of service they require. Examples of such system are *Grid* and *Cloud* computing. Master-working computing (also known as Desktop Grid Computing) exploits the growing use of personal computers and their capabilities (i.e. CPU and GPU) and their high-speed access to the Internet, in providing an even cheaper high performance computing alternative. In particular, personal computing devices all around the world are accessed through the Internet and are used for computations; these devices are called *workers* and the *tasks* (computation jobs) are assigned by a *master* entity (the one that needs the outcome of the tasks' computation). At present, Internet-based master-worker computing is mostly embraced by the scientific community in the form of volunteer computing, where computing resources are volunteered by the public to help solve scientific problems. Among the most popular volunteering projects is SETI@home [2] running on the BOINC [3] platform. A profit-seeking computation platform has also been developed by Amazon, called Mechanical Turk [4][1]. Although the potential is great, the use of master-worker computing is limited by the untrustworthy nature of the workers, who might report incorrect results due to various reasons [3], [5]–[7]. In SETI, the master attempts to minimize the impact of these incorrect results by assigning the same task to several workers and comparing their outcomes (i.e., redundant task allocation is employed [3]).

Prior work, building on redundant task allocation, has considered different approaches in increasing the reliability of master-worker computing [8]–[16]. One such approach is to consider workers to be *rational* [17], [18] in a game-theoretic sense, that is, each worker is selfish and decides whether to truthfully compute and return the correct result or return a bogus result, based on the strategy that best serves its self-interest (increases its benefit). The rationality assumption can conceptually be justified by the work of Shneidman and Parkes [19] where they reason on the connection of rational players (of Algorithmic Mechanism Design) and workers in realistic P2P systems. Several incentive-based algorithmic mechanisms have been devised, e.g., [11]–[14], [20], that employ reward/punish schemes to "enforce" rational workers to act correctly, and hence having the master reliably obtain correct task results. Most of these mechanisms are *one-shot* in the following sense: in a round, the master sends a task to be computed to a collection of workers, and the mechanism,

---

[1]Although in Amazon's Mechanical Turk many tasks are performed by humans, even such cases can be seen as a computational platform, one where processors are indeed humans.

using auditing and reward/punish schemes guarantees (with high probability) that the master gets the correct task result. For another task to be computed, the process is repeated but without taking advantage of the knowledge gained.

The work in [13] takes advantage of the repeated interactions between the master and the workers, by studying the *dynamics of evolution* [21] of such master-worker computations through *reinforcement learning* [22] where both the master and the workers adjust their strategies based on their prior interaction. The objective of the master is to reach a state in the computation after which it always obtains the correct results (called eventual correctness), while the workers attempt to increase their benefit. Roughly speaking, in each round a different task is assigned to the workers, and when the master collects the responses it decides with some probability to verify these answers or not; verification is costly to the master. Each worker decides with some probability to cheat, that is, return a bogus result without computing the task (to save the cost of doing so), or to be honest, that is, truthfully return the correct result. If the master verifies, it then penalizes the cheaters and rewards the honest workers. Also, based on the number of cheaters, it might decide to increase or decrease the probability of verification for the next round. If the master does not verify, then it accepts the result returned by the majority of the workers and rewards this majority (it does not penalize the minority); in this case, it does not change its probability of verifying. Similarly, depending on the payoff received in a given round (reward or punishment minus the cost of performing the task, if it performed it), each worker decides whether to increase or decrease its probability of cheating. It was shown that under certain conditions, eventually workers stop cheating and the master always obtains the correct task results with minimal verification.

Our approach. In this work, we take a different approach in exploiting the repeated interactions between the master and the rational workers. We model this repeated interaction as a *repeated game* [23]. Unlike the work in [13], as long as the workers operate within this framework, the master obtains the correct task results (with high probability) from the very first round. The main idea is the following: when the workers detect that one worker (or more) has deviated from an agreed strategic choice, then they change their strategy into the one that maximizes the negative effect they have on the utility of the deviating worker. This might negatively affect their own utility as well, but in long-running computations (such as master-worker computations) this punishment threat stops workers from deviating from the agreed strategic choice. So, indeed, workers do not deviate. (For more details on the theory of repeated games please refer to [23].) As we demonstrate later, under certain conditions, not only the master obtains the correct results, but in the long run, it does so with lower cost when compared with the repeated use of the one-shot mechanism of [20] or the reinforcement learning mechanism of [13].

Contribution. In summary, the main contributions of this work are the following:

- To the best of our knowledge, this is the first work that attempts to increase the reliability of Internet-based master-worker computing by modeling the repeated interaction between the master and the workers as a repeated game. (The model is formalized in Section II.)
- We first present a mechanism (Section III) where workers decide to cheat or be honest deterministically (in game-theoretic terms they follow pure equilibria strategies), and prove the conditions and the cost under which the master obtains the correct task result in every round (with high probability). In order to allow the workers to detect other workers' deviations (from the agreed strategy), the master needs to provide only the number of different answers received (regardless of whether it verified or not).
- Then, we consider the case where the workers' decision is probabilistic (in game-theoretic terms they follow mixed equilibria strategies). The mechanism (Section IV) in this case is more involved as workers need more information in order to detect deviations. The master provides the workers which answers it has received and how many of each, and the workers use this information to detect deviations. We prove the conditions and the cost under which the master obtains the correct task result in every round (with high probability).
- Finally, we perform a simulation study to demonstrate the utility of our new approach (Section V). The study complements the theoretical analysis by providing more insight on the effectiveness of the mechanisms by experimenting on various parameter values, and also provides comparison with the works in [13] and [20]. In particular, our simulations show that in the presence of $\lceil n/2 \rceil$ deviators (out of n total workers), which is the minimum to have an impact in voting mechanisms, our mechanism performs similarly or better than the reinforcement learning mechanism of [13], and both mechanisms perform significantly better than the repeated use of the one-shot mechanism of [20].

Other related work. A classical Distributing Computing approach for increasing the reliability of master-working computing is to model the malfunctioning (due to a hardware or a software error) or cheating (intentional wrongdoer) as *malicious* workers that wish to hamper the computation and thus always return an incorrect result. The non-faulty workers are viewed as *altruistic* ones [3] that always return the correct result. Under this view, malicious-tolerant protocols have been considered, e.g., [8]–[10], where the master decides on the correct result based on majority voting. More recent works, e.g., [14], [24], have combined this approach with incentive-based game theoretic approaches and devised mechanisms assuming the co-existence of altruistic, malicious and rational workers. The work in [24] employed worker *reputation* to cope with malice, while the work in [14] relied on statistical information on the distribution of the worker types (altruistic, malicious or rational). Extending our present work to cope with malicious workers is an interesting future direction. Mo-

tivated by computation-outsourcing applications, a complexity study of multi-prover rational interactive proofs (MRIP) was presented recently in [25]. In a MRIP, several computationally-unbounded provers communicate with a polynomial-time randomized verifier who wants to determine the membership of an input string to a language. The model in this work is different from ours in several ways. For instance, the provers can pre-agree on how they plan to respond, the aim of the provers is to maximize the *total* rather than individual profits, and the payment-loss of the provers responding incorrectly is smaller if anyway the verifier obtains the correct answer (from others).

II. MODEL

The master-workers framework. We consider a distributed system consisting of a master processor that assigns, over the Internet, computational tasks to a set W of n workers. In particular, the computation is broken into rounds. In each round the master sends a task to be computed to the workers and the workers return the task result. The master, based on the workers' replies, must decide on the value it believes is the correct outcome of the task in the same round. The tasks considered in this work are assumed to have a unique solution; although such limitation reduces the scope of application of the presented mechanism [26], there are plenty of computations where the correct solution is unique: e.g., any mathematical function. In this work security issues are not considered. Security can be achieved by cryptographic means, as done in BOINC [3], which allows for encrypting communication, authenticating master and workers, signing the code of tasks, and executing tasks in sandboxes.

Following [17] and [19], we consider workers to be *rational*, that is, they are selfish in a game-theoretic sense and their aim is to maximize their benefit (utility) under the assumption that other workers do the same. In the context of this paper, a worker is *honest* in a round when it truthfully computes and returns the task result, and it *cheats* when it returns some incorrect value. We denote by p_{Ci}^r the probability of a worker i cheating in round r. Note that we do not consider non-intentional errors produced by hardware or software problems.

To "enforce" workers to be honest, the master employs, when necessary, *verification* and *reward/punish* schemes. The master, in a round, might decide to verify the response of the workers, at a cost. It is assumed that verifying an answer is more efficient than computing the task [27] (e.g., NP-complete problems if $P \neq NP$), but the correct result of the computation is not obtained if the verification fails (e.g., when all workers cheat). We denote by p_V the probability of the master verifying the responses of the workers. The goal of the master is to accept the correct task result in every round, while reducing its utility; therefore, verification needs to be used only when it is necessary. Furthermore, the master can reward and punish workers using the following scheme: When the master verifies, it can accurately reward and punish workers. When the master does not verify, it decides on the majority of the received replies, and it rewards only the majority (and it does not penalize the minority); probability is used to break

symmetry. This is essentially the reward model $\mathcal{R}_m$ on the game $0 : n$ as defined in [12] (also considered in other works, e.g., [13], [14]).

The payoff parameters considered in this work are detailed in Table I. Observe that there are different parameters for the reward WB_A to a worker and the cost MC_A of this reward to the master. This models the fact that the cost to the master might be different from the benefit for a worker.

WP_C	worker's penalty for being caught cheating
WC_T	worker's cost for computing the task
WB_A	worker's benefit from master's acceptance
MP_W	master's penalty for accepting a wrong answer
MC_A	master's cost for accepting the worker's answer
MC_V	master's cost for auditing worker's answers
MB_R	master's benefit from accepting the right answer

TABLE I
PAYOFFS. THE PARAMETERS ARE NON-NEGATIVE.

For the purposes of repeated game framework (presented next), the punishment of the master to a worker caught cheating is proportional to the number of cheaters: Let F denote the set of workers caught cheating in a round that the master verifies. Then, in that round, the master applies penalty $WP_C \cdot |F|$ to *every* worker in F. The fact that punishment is proportional to the number of cheaters is, intuitively, an important tool to implement peer-punishment, which is required in the repeated games framework. Hence, we carry our analysis for proportional punishments. The study of constant punishments is an open question that we leave for future work.

The repeated game framework. We assume that workers participate in the system within the framework of a *repeated game* [23]. The objective of the repeated games model is to capture the effect of long-term interaction. That is, workers take into account that their behavior in one computation will have an effect on the behavior of other workers in the future. We further assume that workers behave according to the reality that they perceive. That is, although their participation is physically bounded to a finite number of rounds, it is known [23] that workers participate in the game as if the number of repetitions is infinite, even for a small number of repetitions, until the last repetition is close. (Note that prediction mechanisms such as the one in [28], can be used to establish the availability of a set of workers for a sufficiently long period of time.) In this paper, we assume that workers participate unaware of when their participation will end, and we analyze the system as an *infinite repeated game*. In the following game specification, we follow the notation in [23].

Let the set of workers be W, and for each worker $i \in W$ let the set of strategies cheat and not-cheat be $A_i = \{C, \overline{C}\}$, and the preference relation (which is the obvious preference for a higher worker utility) be $u_i : A \to \mathbb{R}$, $A = \times_{j \in W} A_j$. Then, we model each round of computation as the mixed extension $G' = \langle W, (\Delta(A_i))_{i \in W}, (U_i)_{i \in W} \rangle$ of the strategic game $G = \langle W, (A_i)_{i \in W}, (u_i)_{i \in W} \rangle$, where $\Delta(A_i)$ is the set of

probability distributions over A_i, and $U_i : \times_{j \in W} \Delta(A_j) \to \mathbb{R}$ is the expected utility of worker i under u_i induced by $\times_{j \in W} \Delta(A_j)$.

On the other hand, we model the multiple-round long-running computation as an infinitely repeated game where the constituent game is G'. That is, an extensive game with simultaneous moves $\langle W, H, P, (U_i^*)_{i \in W} \rangle$, where $H = \{\emptyset\} \cup (\cup_{r=1}^{\infty} A^r) \cup A^{\infty}$, $\emptyset$ is the initial empty history, A^{∞} is the set of infinite sequences $(a^r)_{r=1}^{\infty}$ of action profiles in G, the (next) player function is $P(h) = W$ (simultaneous moves) for each history $h \in H$, and U_i^* is a preference relation on A^{∞} that extends the preference relation U_i to infinite rounds under the limit of means criterion. That is, the payoff of all rounds is evaluated symmetrically, in contrast with other criteria where the value of a given gain may diminish with time.

We are interested in equilibria that harness long-term objectives of the workers, rather than strategies that apply to short-sighted workers that would isolate each round of computation as a single game. To support equilibria of the infinitely repeated game that are not simple repetitions of equilibria in the constituent game, workers are deterred from deviating by being punished by other workers. Specifically, any deviation from an agreed equilibrium, called a *trigger strategy*, of some worker i is punished by all other workers changing their strategy to enforce the *minmax payoff*[2] of v_i, that is, the lowest payoff that the other workers can force upon worker i.

Any payoff profile w, where for each worker it is $w_i \geq v_i$, is called *enforceable*. The Nash folk theorem for the limit of means criterion [23] establishes that every feasible[3] enforceable payoff profile of the constituent game is a Nash equilibrium payoff profile of the limit of means infinitely repeated game. Thus, any trigger strategy that yields an enforceable payoff profile is an equilibrium. So, we focus our effort in finding the minmax payoff to later analyze which of the infinitely many trigger criteria yields a mechanism that is beneficial for the master and the workers. For clarity, we present our mechanisms punishing the deviant indefinitely. Nevertheless, it is enough to held the deviant's payoff to the minmax level for enough rounds to wipe out its gain from the deviation, as shown by the Perfect folk theorem [23].

In order to make punishment decisions, workers need information about previous outcomes. We consider two scenarios, one where the master only provides the number of different answers received, and another where the master informs which are the answers received, and how many of each. In the first case, workers are bounded to use pure equilibria of the constituent game, so that non-deviators can decide to punish if there were workers replying with an answer other than theirs. For the second case, a worker that did not cheat may count the number of answers different from its own. But a worker that cheated needs to know which one was correct. We assume that workers can verify the answers with negligible cost. Notice that the same does not apply to the master, which is assumed to

have a high operation cost yielding a high cost of verification.

III. LIMITED INFORMATION (PURE EQUILIBRIA)

In this section we assume that workers decide to be honest or to cheat deterministically, i.e., follow pure strategies. Under this assumption we define the mechanism shown in Algorithm 2. We will show using repeated-games analysis that this mechanism leads to an equilibrium where no worker cheats. The algorithm defined requires in every round the number of different answers received by the master in the round. This is provided by the master, as shown in Algorithm 1.

Algorithm 1: Pure strategies master algorithm. The probability of verification $p_{\mathcal{V}}$ is any value such that $WC_{\mathcal{T}}/(WB_{\mathcal{A}} + WP_{\mathcal{C}}\lceil n/2 \rceil) < p_{\mathcal{V}} < (WB_{\mathcal{A}} + WC_{\mathcal{T}})/(2WB_{\mathcal{A}} + nWP_{\mathcal{C}})$

---

1 **while** *true* **do**
2   send a computational task to all the workers in W
3   **upon** *receiving all answers* **do**
4   with probability $p_{\mathcal{V}}$, verify the answers
5   **if** *the answers were not verified* **then** accept the majority
6   reward/penalize accordingly
7   send to the workers the number of different answers received

---

Algorithm 2: Pure strategies worker algorithm.

---

1 strategy $\leftarrow \overline{\mathcal{C}}$
2 **while** *true* **do**
3   **upon** *receiving a task* **do**
4   **if** *strategy*$= \overline{\mathcal{C}}$ **then**
5   compute the task and send the result to the master
6   **else**
7   do not compute and send a bogus result to the master
8   **upon** *receiving from the master the number of different answers* **do**
9   **if** *number of different answers* > 1 **then**
10   strategy $\leftarrow \mathcal{C}$

We analyze now the properties of this mechanism. Although the algorithms allow that a cheating worker replies with any value, in the analysis of the rest of the section we assume that all workers that cheat in a round return the same incorrect value (as done also, for example, in [8], [13]). Intuitively, this assumption yields a worst case scenario (and hence analysis) for the master with respect to obtaining the correct result.

Recall that, for any given round of the constituent game, U_i is the expected utility of worker i and s_i is the strategy chosen by worker i. Let F be the set of cheaters and $F_{-i} = \{j | j \in W \wedge j \neq i \wedge s_j = \mathcal{C}\}$. Then, the following holds.

Lemma 1: If $WB_{\mathcal{A}} > WC_{\mathcal{T}}$ and $WP_{\mathcal{C}} > WC_{\mathcal{T}}$, for any $p_{\mathcal{V}}$ such that $WC_{\mathcal{T}}/(WB_{\mathcal{A}} + WP_{\mathcal{C}}\lceil n/2 \rceil) < p_{\mathcal{V}} < (WB_{\mathcal{A}} + WC_{\mathcal{T}})/(2WB_{\mathcal{A}} + nWP_{\mathcal{C}})$, and for any worker $i \in W$, the minmax expected payoff is $v_i = (1 - p_{\mathcal{V}})WB_{\mathcal{A}} - p_{\mathcal{V}}nWP_{\mathcal{C}}$, which is obtained when $s_i = \mathcal{C}$ and $|F_{-i}| > (n-1)/2$.

Proof: We notice first that, given that $WB_\mathcal{A} > WC_\mathcal{T}$ and $WP_\mathcal{C} > WC_\mathcal{T}$, the range of values $WC_\mathcal{T}/(WB_\mathcal{A} + WP_\mathcal{C}\lceil n/2 \rceil) < p_\mathcal{V} < (WB_\mathcal{A} + WC_\mathcal{T})/(2WB_\mathcal{A} + nWP_\mathcal{C})$ for $p_\mathcal{V}$ is not empty. For any worker i, there are four possible utility outcomes, namely.

$$U_i(s_i = \mathcal{C}, |F| > n/2) = (1 - p_\mathcal{V})WB_\mathcal{A} - p_\mathcal{V}WP_\mathcal{C}|F|, \quad (1)$$
$$U_i(s_i = \overline{\mathcal{C}}, |F| > n/2) = p_\mathcal{V}WB_\mathcal{A} - WC_\mathcal{T}, \quad (2)$$
$$U_i(s_i = \mathcal{C}, |F| < n/2) = -p_\mathcal{V}WP_\mathcal{C}|F|, \quad (3)$$
$$U_i(s_i = \overline{\mathcal{C}}, |F| < n/2) = WB_\mathcal{A} - WC_\mathcal{T}. \quad (4)$$

We want to find worker i's minmax payoff, which is the lowest payoff that the other workers can force upon i (cf. [23]). That is,

$$v_i = \min_{s_{-i} \in \{\mathcal{C}, \overline{\mathcal{C}}\}^{n-1}} \max_{s_i \in \{\mathcal{C}, \overline{\mathcal{C}}\}} U_i(s_i, s_{-i}).$$

From the perspective of worker i, the actions of the remaining workers fall in one of three cases: $|F_{-i}| < (n-1)/2$, $|F_{-i}| = (n-1)/2$, and $|F_{-i}| > (n-1)/2$. Thus, we have

$$v_i = \min\{ \max_{s_i \in \{\mathcal{C}, \overline{\mathcal{C}}\}} U_i(s_i, |F_{-i}| < (n-1)/2),$$
$$\max_{s_i \in \{\mathcal{C}, \overline{\mathcal{C}}\}} U_i(s_i, |F_{-i}| = (n-1)/2),$$
$$\max_{s_i \in \{\mathcal{C}, \overline{\mathcal{C}}\}} U_i(s_i, |F_{-i}| > (n-1)/2)\}. \quad (5)$$

From Equations 3 and 4, and given that $WB_\mathcal{A} > WC_\mathcal{T}$,

$$\max_{s_i \in \{\mathcal{C}, \overline{\mathcal{C}}\}} U_i(s_i, |F_{-i}| < (n-1)/2)$$
$$= \max\{-p_\mathcal{V}WP_\mathcal{C}|F|, WB_\mathcal{A} - WC_\mathcal{T}\}$$
$$= WB_\mathcal{A} - WC_\mathcal{T}, \text{ for } 1 \le |F| < \lceil n/2 \rceil. \quad (6)$$

From Equations 1 and 2, and given that $p_\mathcal{V} < (WB_\mathcal{A} + WC_\mathcal{T})/(2WB_\mathcal{A} + nWP_\mathcal{C})$, it is

$$\max_{s_i \in \{\mathcal{C}, \overline{\mathcal{C}}\}} U_i(s_i, |F_{-i}| > (n-1)/2)$$
$$= \max\{(1 - p_\mathcal{V})WB_\mathcal{A} - p_\mathcal{V}WP_\mathcal{C}|F|, p_\mathcal{V}WB_\mathcal{A} - WC_\mathcal{T}\},$$
$$= (1 - p_\mathcal{V})WB_\mathcal{A} - p_\mathcal{V}WP_\mathcal{C}|F|, \text{ for } \lceil n/2 \rceil < |F| \le n. \quad (7)$$

From Equations 1 and 4, and given that $p_\mathcal{V} > WC_\mathcal{T}/(WB_\mathcal{A} + WP_\mathcal{C}\lceil n/2 \rceil)$, it is

$$\max_{s_i \in \{\mathcal{C}, \overline{\mathcal{C}}\}} U_i(s_i, |F_{-i}| = (n-1)/2)$$
$$= \max\{(1 - p_\mathcal{V})WB_\mathcal{A} - p_\mathcal{V}WP_\mathcal{C}\lceil n/2 \rceil, WB_\mathcal{A} - WC_\mathcal{T}\}$$
$$= WB_\mathcal{A} - WC_\mathcal{T}. \quad (8)$$

Replacing Equations 6, 7, and 8 in Equation 5, we obtain for $\lceil n/2 \rceil < |F| \le n$

$$v_i = \min\{WB_\mathcal{A} - WC_\mathcal{T}, (1 - p_\mathcal{V})WB_\mathcal{A} - p_\mathcal{V}WP_\mathcal{C}|F|\},$$
$$= (1 - p_\mathcal{V})WB_\mathcal{A} - p_\mathcal{V}nWP_\mathcal{C}.$$

The latter is true because $p_\mathcal{V} > WC_\mathcal{T}/(WB_\mathcal{A} + WP_\mathcal{C}\lceil n/2 \rceil) > WC_\mathcal{T}/(WB_\mathcal{A} + nWP_\mathcal{C})$. Thus, the claim holds. ∎

The following theorem establishes the correctness of our pure-strategies mechanism. The proof follows from Lemma 1 and the repeated-games framework [23].

Theorem 2: For a long-running multi-round computation system with set of workers W, and for a set of payoff parameters such that $WB_\mathcal{A} > WC_\mathcal{T}$ and $WP_\mathcal{C} > WC_\mathcal{T}$, the mechanism defined in Algorithms 1 and 2 guarantees that the master always obtains the correct answer. In each round, the utility of each worker i is $U_i = WB_\mathcal{A} - WC_\mathcal{T}$ and the expected utility of the master is $U_M = MB_\mathcal{R} - nMC_\mathcal{A} - p_\mathcal{V}MC_\mathcal{V}$.

IV. More Information Helps (mixed equilibria)

In this section, we consider the general case in which workers can randomize their decision to be honest of cheat. In this case, the equilibrium in the constituent game is mixed, that is, in the equilibrium workers cheat with some probability $p \in (0, 1)$, rather than behaving deterministically. Hence, the actual probability used by each worker cannot be inferred accurately from the outcome of one computation. In other words, even knowing that some given worker has cheated after a computation, it is not possible to know if this event was a deviation from the equilibrium or not from one single worker outcome. Nonetheless, it is possible to provide stochastic guarantees, either from many computations of one worker, or one computation by multiple workers. Such guarantees may be enough for some scenarios.

Specifically, if the master announces how many of each answer has received, workers may make punishment decisions based on the probability of such outcome. This is possible even for workers that did not compute the task, given that the cost of verification for workers is assumed to be negligible. Such decision will not be based on deterministic information, but we can provide guarantees on the probability of being the right decision. Hence, in this section we define a mechanism where the master sends to all the workers in each round the answers that it has received and how many of each. This is described in Algorithm 3. In summary, the approach is to carry out the computation as in a regular repeated game, but punishments are applied when it is known that some workers have deviated from the equilibrium of the constituent game with some parametric probability. The mechanism that is assigned to the workers is described in Algorithm 4.

We emphasize that what is punished by peer-workers is deviation from the agreed equilibrium (rather than cheating which is punished by the master). That is, if the equilibrium is to cheat with some probability p and the number of incorrect answers was x, punishment is applied when the probability of x incorrect answers is very low if all workers were using p, either if x is less or more than the number of cheaters expected.

It is known [23] that it is enough to apply peer-punishment for a number of rounds that neutralizes the benefit that the deviators might have obtained by deviating. Nonetheless, to avoid unnecessary clutter, we omit this detail in Algorithm 4 where punishment is applied forever. In our simulations in Section V, we limit the punishment to one round, since that

Algorithm 3: Mixed strategies master algorithm. $p_\mathcal{V}$ is set according to Lemma 5.

---
1 while *true* **do**
2   send a computational task to all the workers in W
3   **upon** *receiving all answers* **do**
4   with probability $p_\mathcal{V}$, verify the answers
5   **if** *the answers were not verified* **then** accept the majority
6   reward/penalize accordingly
7   send to the workers a list of pairs $\langle$answer, count$\rangle$

---

is enough to neutralize the benefit of a deviation for those parameters.

Algorithm 4: Mixed strategies worker algorithm. $p_\mathcal{C}$ is initialized according to Lemma 5. $\varepsilon > 0$ is the probability of erroneous punishment (cf. Lemma 3).

---
1 maxrounds $\leftarrow \lfloor 3np_\mathcal{C} \ln(1/\varepsilon)\rfloor$ // Punishment decisions only for $\delta \geq 1/np_\mathcal{C}$ (cf. Lemma 3).
2 counts $\leftarrow$ empty queue of integers   // counts[i] is the $(i+1)$th item, for $i = 0,1,2,\ldots$.
3 for *each* round $= 1,2,\ldots$ **do**
4   **upon** *receiving a task* **do**
   // computation phase
5   cheat $\leftarrow \begin{cases} true, & \text{with probability } p_\mathcal{C} \\ false, & \text{with probability } 1-p_\mathcal{C} \end{cases}$
6   **if** cheat $= false$ **then** result $\leftarrow$ task result computed
7   **else** result $\leftarrow$ bogus result
8   send result to the master
   // punishment phase
9   **upon** *receiving from the master a list of pairs* $\langle$answer, count$\rangle$ **do**
    // update # of cheaters per round
10   verify all answers
11   #incorrect $\leftarrow$ number of incorrect answers
12   enqueue #incorrect to counts
13   **if** *size of* counts $>$ maxrounds **then** dequeue from counts
    // punishment decision
14   cheaters$_{\min} \leftarrow n$
15   cheaters$_{\max} \leftarrow 0$
16 $R \leftarrow \min\{$maxrounds, round$\}$
17   **for** $r = 1$ to R **do**
18   **if** counts[$R-r$] $<$ cheaters$_{\min}$ **then** cheaters$_{\min} \leftarrow$ counts[$R-r$]
19   **if** counts[$R-r$] $>$ cheaters$_{\max}$ **then** cheaters$_{\max} \leftarrow$ counts[$R-r$]
20 $\delta \leftarrow \sqrt{3\ln(1/\varepsilon)/(rnp_\mathcal{C})}$
21   **if** $\delta < 1$ **then**
22   **if** cheaters$_{\min} \geq \lceil(1+\delta)np_\mathcal{C}\rceil$ **or** cheaters$_{\max} \leq \lfloor(1-\delta)np_\mathcal{C}\rfloor$ **then**
       // Lemma 3
23 $p_\mathcal{C} \leftarrow 1$ // Lemma 4

As in the analysis of the previous section, to obtain a worse case analysis for the master, we assume that cheating workers return the same incorrect result.

The following lemma characterizes the number of incorrect answers and the number of rounds that should trigger a punishment.

Lemma 3: In a system with n workers where the mixed equilibrium of the constituent game is to cheat with some probability $p_\mathcal{C} > 0$, if the number of incorrect answers is at least $(1+\delta)\lceil np_\mathcal{C}\rceil$ or at most $(1-\delta)\lfloor np_\mathcal{C}\rfloor$ during r consecutive rounds, for any $1/(np_\mathcal{C}) \leq \delta < 1$, $r \geq 1$, and $\varepsilon > 0$ such that $r\delta^2 > 3\ln(1/\varepsilon)/(np_\mathcal{C})$, then there are one or more workers deviating from that equilibrium with probability at least $1-\varepsilon$.

Proof: Let $1, 2, \ldots, n$ be some labeling identifying the workers. For any given round of computation, let X_i be a random variable indicating whether worker i cheated or not, and let $X = \sum_{i=1}^{n} X_i$. For $i = 1, 2, \ldots, n$, the X_i random variables are not correlated. Thus, we can upper bound the tails of the probability distribution on the number of cheaters using the following Chernoff-Hoeffding bounds [29]. For $0 < \delta \leq 1$, it is $Pr(X \geq (1+\delta)np_\mathcal{C}) \leq e^{-np_\mathcal{C}\delta^2/3}$ and for $0 < \delta < 1$, it is $Pr(X \leq (1-\delta)np_\mathcal{C}) \leq e^{-np_\mathcal{C}\delta^2/2}$. Therefore, it is $Pr(X \geq (1+\delta)\lceil np_\mathcal{C}\rceil) \leq e^{-np_\mathcal{C}\delta^2/3}$ and $Pr(X \leq (1-\delta)\lfloor np_\mathcal{C}\rfloor) \leq e^{-np_\mathcal{C}\delta^2/2}$. Given that X cannot differ from the expected number of cheaters by less than one worker, we further restrict δ as follows. For $1/(np_\mathcal{C}) \leq \delta < 1$, it is $Pr(X \geq (1+\delta)\lceil np_\mathcal{C}\rceil) \leq e^{-np_\mathcal{C}\delta^2/3}$ and $Pr(X \leq (1-\delta)\lfloor np_\mathcal{C}\rfloor) \leq e^{-np_\mathcal{C}\delta^2/2}$.

Letting E_{high} be the event of having $X \geq (1+\delta)\lceil np_\mathcal{C}\rceil$ incorrect answers for r consecutive rounds, and E_{low} be the event of having $X \leq (1-\delta)\lfloor np_\mathcal{C}\rfloor$ incorrect answers for r consecutive rounds, it is $Pr(E_{high}) \leq e^{-nrp_\mathcal{C}\delta^2/3}$ and it is $Pr(E_{low}) \leq e^{-nrp_\mathcal{C}\delta^2/2}$. Given that $e^{-nrp_\mathcal{C}\delta^2/2} \leq e^{-nrp_\mathcal{C}\delta^2/3} \leq \varepsilon$ for $r\delta^2 \geq 3\ln(1/\varepsilon)/(np_\mathcal{C})$, if either E_{high} or E_{low} occurs, there are one or more workers deviating from equilibrium with probability at least $1-\varepsilon$ and the claim follows. ∎

In the following lemma, we show what is the minmax payoff when a mixed equilibrium of the constituent game is allowed. The proof is left to the Appendix due to space restrictions.

Lemma 4: For any $p_\mathcal{V}$, such that $p_\mathcal{V} > 2WB_\mathcal{A}/(2WB_\mathcal{A} + WP_\mathcal{C}n)$ and $p_\mathcal{V} \geq WC_\mathcal{T}/WB_\mathcal{A}$, and for any worker $i \in W$, the minmax expected payoff of worker i is $v_i = p_\mathcal{V}WB_\mathcal{A} - WC_\mathcal{T}$, which is attained when all the other workers use $p_\mathcal{C} = 1$.

Given that the aim of the mechanism is to provide correctness, $p_\mathcal{V}$ must be lower bounded and $p_\mathcal{C}$ upper bounded to enforce an equilibrium that provides correctness guarantees in each computation with parametric probability, which we do in the following lemma.

Lemma 5: In any given round of computation, and for any $\varphi > 0$ and $0 < \xi \leq 1$, if each worker cheats with probability $p_\mathcal{C} < 1/(2(1+\xi))$ and the master verifies with probability $p_\mathcal{V} \geq (e^{-n\xi^2/(6(1+\xi))} - \varphi)/(e^{-n\xi^2/(6(1+\xi))} - p_\mathcal{C}^n)$, the probability that the master obtains a wrong answer is at most φ.

Proof: Let $1, 2, \ldots, n$ be some labeling identifying the workers. Let X_i be a random variable indicating whether worker i cheated or not, and let $X = \sum_{i=1}^{n} X_i$. For $i = 1, 2, \ldots, n$, the X_i random variables are not correlated. Thus, we can upper bound the probability of having a majority of

cheaters using the following Chernoff-Hoeffding bound [29]. For $0 < \xi \leq 1$, it is $Pr(X \geq (1+\xi)np_{\mathcal{C}}) \leq e^{-np_C\xi^2/3}$. For $p_{\mathcal{C}} < 1/(2(1+\xi))$ we get

$$Pr(X \geq (1+\xi)np_{\mathcal{C}}) \leq e^{-np_C\xi^2/3}$$
$$Pr(X > n/2) \leq e^{-n\xi^2/(6(1+\xi))}.$$

Thus, for the master to achieve correctness with probability at least $1 - \varphi$, it is enough to have $(1 - p_{\mathcal{V}})e^{-n\xi^2/(6(1+\xi))} + p_{\mathcal{V}}p_{\mathcal{C}}^n \leq \varphi$, which is true for $p_{\mathcal{V}} \geq \frac{e^{-n\xi^2/(6(1+\xi))}-\varphi}{e^{-n\xi^2/(6(1+\xi))}-p_{\mathcal{C}}^n}$ if $p_{\mathcal{C}} < 1/e^{\xi^2/(6(1+\xi))}$, which holds for $p_{\mathcal{C}} < 1/(2(1+\xi))$. ∎

The following theorem establishes the correctness guarantees of our mixed-strategies mechanism.

Theorem 6: Consider a long-running multi-round computation system with set of workers such that $n > 2$. For any set of payoff parameters, and any $0 < p_{\mathcal{C}} < 1/(2(1+\xi))$ for some $0 < \xi \leq 1$, setting the probability of verification so that $p_{\mathcal{V}} > 2\,WB_{\mathcal{A}}/(2\,WB_{\mathcal{A}} + WP_{\mathcal{C}}n)$, $p_{\mathcal{V}} \geq WC_{\mathcal{T}}/WB_{\mathcal{A}}$, and $p_{\mathcal{V}} \geq \frac{e^{-n\xi^2/(6(1+\xi))}-\varphi}{e^{-n\xi^2/(6(1+\xi))}-p_{\mathcal{C}}^n}$, for some $\varphi > 0$, the following applies to each round of computation. If workers comply with the repeated games framework when punishment is stochastically consistent, the mechanism defined in Algorithms 3 and 4 guarantees that the master obtains the correct answer with probability at least $1 - \varphi$, the expected utility of the master is

$$U_M = p_{\mathcal{V}}(MB_{\mathcal{R}} - MC_{\mathcal{V}} + np_{\mathcal{C}} WP_{\mathcal{C}} - n(1-p_{\mathcal{C}})MC_{\mathcal{A}})$$
$$+ (1 - p_{\mathcal{V}})\Big(\sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{i}p_{\mathcal{C}}^i(1-p_{\mathcal{C}})^{n-i}\,(MB_{\mathcal{R}} - iMC_{\mathcal{A}})$$
$$- \sum_{i=\lceil n/2 \rceil}^{n} \binom{n}{i}p_{\mathcal{C}}^i(1-p_{\mathcal{C}})^{n-i}\,(MP_{\mathcal{W}} + iMC_{\mathcal{A}})\Big),$$

and the expected utility of each worker i is

$$U_i = (p_{\mathcal{C}}(1-p_{\mathcal{V}})p_{\geq h} + (1-p_{\mathcal{C}})p_{\mathcal{V}}p_{>h} + (1-p_{\mathcal{C}})p_{\leq h})\,WB_{\mathcal{A}}$$
$$- p_{\mathcal{C}}p_{\mathcal{V}}(1 + (n-1)p_{\mathcal{C}})\,WP_{\mathcal{C}} - (1-p_{\mathcal{C}})WC_{\mathcal{T}}.$$

Where

$$p_{\geq h} = \sum_{j=(n-1)/2}^{n-1} \binom{n-1}{j}p_{\mathcal{C}}^j(1-p_{\mathcal{C}})^{n-1-j},$$
$$p_{>h} = \sum_{j=(n+1)/2}^{n-1} \binom{n-1}{j}p_{\mathcal{C}}^j(1-p_{\mathcal{C}})^{n-1-j},\ \text{and}$$
$$p_{\leq h} = \sum_{j=0}^{(n-1)/2} \binom{n-1}{j}p_{\mathcal{C}}^j(1-p_{\mathcal{C}})^{n-1-j}.$$

Proof: First, we notice that by making $p_{\mathcal{C}}$ arbitrarily close to 0, the expected worker utility is arbitrarily close to $WB_{\mathcal{A}} - WC_{\mathcal{T}}$, which is bigger than the minmax payoff $v_i = p_{\mathcal{V}}WB_{\mathcal{A}} - WC_{\mathcal{T}}$ for any $p_{\mathcal{V}} < 1$. Then, there exists an enforceable payoff profile w, that is, a payoff profile where $w_i \geq v_i$ for each worker i and, hence, there exists a Nash equilibrium payoff profile that all workers will follow due to long-term rationality (cf. Proposition 144.3 in [23]). The rest of the claim follows from Lemmas 3, 4, and 5. ∎

V. SIMULATIONS

A. Design

In this section, we present our simulations of the mechanism in Algorithms 3 and 4. For the sake of contrast, we also carry out simulations for the mechanism in [13], and for a repeated application of the one-shot mechanism in [20]. For easy reference, we denote our mechanism as *RG* (repeated games), the mechanism in [13] as *ED* (evolutionary dynamics), the mechanism in [20] as *OS* (one shot), and the repeated application of OS in a multi-round computation environment as *ROS* (repeated one shot).

These mechanisms differ as follows. RG includes the threat of a peer-punishment that stops workers from deviating from the agreed equilibrium, ED is aimed to converge to correctness after some time (rounds), and OS does not include previsions for equilibrium deviations. The common assumption among all three approaches is that workers comply with the mechanism laid out. In RG, workers are assumed to be rational in the long term, and consequently they will agree on a given Nash Equilibrium (i.e., a $p_{\mathcal{C}}$) that is computed taking into account the potential profit of future interactions. In ED, it is assumed that workers update their $p_{\mathcal{C}}$ for the next computation round using a particular formula, which is a function of the $p_{\mathcal{C}}$ and the payment received in the previous computation round. In OS, workers are assumed to be rational but short-sighted, that is, they do not take into account future interactions (short-term rationality).

Once the parameters (payment, punishment, etc.) have been fixed (as we do for simulations), the performance of RG, ED, and ROS when all workers comply with the agreed protocol can be compared by a simple computation of expected utilities and probability of correctness. However, the tradeoffs between correctness and utilities when worker misbehavior occurs (in other words, what is the resilience of these systems to deviations) is an open question that we answer experimentally as follows.

Table II summarizes the parameter values used in our simulations and Table III shows the master and workers' utilities as derived under the specific parameter values. From the simulation results for 27 and 81 workers (not shown for space limitation), conclusions similar to those obtained for 9 workers can be derived.

For the sake of fair comparison, we use for all three mechanisms the same parameters used in [13], [20] whenever possible (summarized in Table II). Specifically, we set $WC_{\mathcal{T}} = 0.1$. For each $n \in \{9, 27, 81\}$ and $WB_{\mathcal{A}} \in \{1, 1.1, 1.2, \ldots, 2\}$, we set $WP_{\mathcal{C}} = WB_{\mathcal{A}}/(np_{\mathcal{C}})$ for RG. That is, the expected punishment in RG is $WB_{\mathcal{A}}$. Given that in ED and ROS the worker punishment is a constant value (i.e., it is not proportional to the number of cheaters), to simulate ED and OS we set $WP_{\mathcal{C}} = WB_{\mathcal{A}}$, that is, the expected punishment in RG is $WB_{\mathcal{A}}$. As indicated in [13], these parameters are consistent with statistics obtained in SETI@home projects.

Observing the conditions of Theorem 6, for RG we fix $p_{\mathcal{C}} = 0.1 < 1/(2(1+\xi))$ for any $\xi > 0$, and we fix

	RG [this paper]	ROS [20]	ED [13]
n	$\{9, 27, 81\}$	$\{9, 27, 81\}$	$\{9, 27, 81\}$
$WB_{\mathcal{A}}$	$\{1, 1.1, \ldots, 2\}$	$\{1, 1.1, \ldots, 2\}$	$\{1, 1.1, \ldots, 2\}$
$WP_{\mathcal{C}}$	$\frac{WB_{\mathcal{A}}}{np_{\mathcal{C}}}$ if $|F| < n$, 0 if $|F| = n$	$WB_{\mathcal{A}}$	$WB_{\mathcal{A}}$
$WC_{\mathcal{T}}$	0.1	0.1	0.1
$p_{\mathcal{C}}$	$\lfloor n/2 \rfloor$:0.1, $\lceil n/2 \rceil$: $\{0.5, 0.6, \ldots, 1\}$	$\lfloor n/2 \rfloor$:0, $\lceil n/2 \rceil$: $\{0.5, 0.6, \ldots, 1\}$	$\lfloor n/2 \rfloor$:0, $\lceil n/2 \rceil$: $\{0.5, 0.6, \ldots, 1\}$
$p_{\mathcal{V}}$	0.17	$\frac{WB_{\mathcal{A}}+0.1}{3\,WB_{\mathcal{A}}} + 0.01$	initially: 0.5, min: 0.01
$MC_{\mathcal{A}}$	$WB_{\mathcal{A}}$	$WB_{\mathcal{A}}$	$WB_{\mathcal{A}}$
$MC_{\mathcal{V}}$	$n\,WB_{\mathcal{A}}$	$n\,WB_{\mathcal{A}}$	$n\,WB_{\mathcal{A}}$
$MP_{\mathcal{W}}$	$n\,WB_{\mathcal{A}}$	$n\,WB_{\mathcal{A}}$	$n\,WB_{\mathcal{A}}$
$MB_{\mathcal{R}}$	$n\,WB_{\mathcal{A}}$	$n\,WB_{\mathcal{A}}$	$n\,WB_{\mathcal{A}}$
other	$\varepsilon = 0.01$	–	$\tau = 0.5$, $a_w = 0.1$, $\alpha_m = \alpha_w = 0.01$

TABLE II

SIMULATIONS PARAMETERS.

	RG [this paper]	ROS [20]	ED [13]
$U_i(\text{verified}, s_i = \mathcal{C})$	$WB_{\mathcal{A}} - WC_{\mathcal{T}}$		
$U_i(\text{verified}, s_i = \overline{\mathcal{C}})$	$-WP_{\mathcal{C}}|F|$	$-WP_{\mathcal{C}}$	
$U_i(\text{not verified}, s_i = \mathcal{C}, |F| > n/2)$	$WB_{\mathcal{A}}$		
$U_i(\text{not verified}, s_i = \overline{\mathcal{C}}, |F| > n/2)$	$-WC_{\mathcal{T}}$		
$U_i(\text{not verified}, s_i = \mathcal{C}, |F| < n/2)$	0		
$U_i(s_i = \overline{\mathcal{C}}, |F| < n/2)$	$WB_{\mathcal{A}} - WC_{\mathcal{T}}$		
$U_M(\text{verified}, |F| < n)$	$MB_{\mathcal{R}} - MC_{\mathcal{V}} - (n-|F|)MC_{\mathcal{A}} + |F|^2 WP_{\mathcal{C}}$	$MB_{\mathcal{R}} - MC_{\mathcal{V}} - (n-|F|)MC_{\mathcal{A}} + |F|WP_{\mathcal{C}}$	
$U_M(\text{verified}, |F| = n)$	$-MC_{\mathcal{V}} + n^2 WP_{\mathcal{C}}$		$MB_{\mathcal{R}} - MC_{\mathcal{V}} + nWP_{\mathcal{C}}$
$U_M(\text{not verified}, |F| > n/2)$	$-MP_{\mathcal{W}} - |F|MC_{\mathcal{A}}$		
$U_M(\text{not verified}, |F| < n/2)$	$MB_{\mathcal{R}} - (n-|F|)MC_{\mathcal{A}}$		

TABLE III

MASTER AND WORKER UTILITIES.

$p_{\mathcal{V}} = 0.17$ which for the parameters chosen verifies that $p_{\mathcal{V}} > 2\,WB_{\mathcal{A}}/(2\,WB_{\mathcal{A}} + WP_{\mathcal{C}}n) = 1/6$, that $p_{\mathcal{V}} \geq WC_{\mathcal{T}}/WB_{\mathcal{A}}$ since $WC_{\mathcal{T}}/WB_{\mathcal{A}} \leq 1/10$, and that $p_{\mathcal{V}} \geq (e^{-n\xi^2/(6(1+\xi))} - \varphi)/(e^{-n\xi^2/(6(1+\xi))} - p_{\mathcal{C}}^n)$, which is true for some $\varphi > 0$ and $\xi > 0$ as required by Theorem 6. To implement the punishment decision in RG, we set the probability of punishment error to $\varepsilon = 0.01$

For ROS, we set $p_{\mathcal{V}} = (WB_{\mathcal{A}} + 0.1)/(3\,WB_{\mathcal{A}}) + 0.01 > (WB_{\mathcal{A}} + WC_{\mathcal{T}})/(WP_{\mathcal{C}} + 2\,WB_{\mathcal{A}})$ as required by OS. For the master payoffs, the aim in [13] was to focus on the master cost making $MC_{\mathcal{V}} = 20$ but zeroing $MP_{\mathcal{W}}$ and $MB_{\mathcal{R}}$ to exclude the impact of whether the correct answer is obtained or not. Here, we consider such impact in all three mechanisms making $MC_{\mathcal{V}} = MP_{\mathcal{W}} = MB_{\mathcal{R}} = n\,WB_{\mathcal{A}}$, and we set $MC_{\mathcal{A}} = WB_{\mathcal{A}}$ assuming that the master cost of accepting an answer is just the payment that the worker receives (no overheads).

It is fair to notice that in ED the master checks the answers received by computing the task itself (*audit*), which is usually more costly than just verifying a given solution (e.g., all NP problems that are not in P). Moreover, when the master audits, it obtains the correct result even if all workers cheat, which is not the case when verifying. Given that the master is penalized for obtaining a wrong answer, when verifying its utility is negatively affected with respect to auditing when no punishment is received by the master. For these simulations, we maintain the same value for the master cost of verification or auditing, but in our model we zero the master punishment when all workers cheat (cf. Table III).

We assume that worker deviations occur in 0.5% of the computation rounds, and that these deviations continue until "fixed" (if possible). That is, we evaluate the performance of all three mechanisms for 200 rounds of computation introducing an identical initial perturbation in them. Specifically, we set $\lfloor n/2 \rfloor$ workers to start with $p_{\mathcal{C}} = 0$ in ED and ROS, and $p_{\mathcal{C}}$ as defined above for RG. For the remaining $\lceil n/2 \rceil$ workers, we evaluate the range $p_{\mathcal{C}} \in \{0.5, 0.6, \ldots, 1\}$ in all

three systems. Notice that this number of deviators is minimal to have an impact on voting schemes. ED will make the deviators converge to $p_\mathcal{C} = 0$ by design. ROS does not include previsions for deviators so they will not return to the desired behavior. For RG, we assume that after being punished by peers in one round, workers return to the agreed equilibrium in the following round. Under such conditions, we compute the number of rounds when the master obtains the correct answer, the master and worker utilities aggregated over all these rounds, and we measure the convergence time for ED and the time to detect the deviation for RG.

B. Discussion

The results of our simulations for $n = 9$ are shown in Figure 1. Similar results for $n = 27$ and 81 are presented in the full version of this work [30]. The results shown correspond to one execution of the simulator. Multiple executions were carried out obtaining similar results.

It can be seen in Figure 1.(a) that the number of rounds when the master obtained the correct answer is similar for RG and ED, except when the deviator $p_\mathcal{C}$ comes closer to 1 where the performance of ED worsens. In general, both mechanisms achieve significantly better correctness than ROS. With respect to utilities, Figure 1.(b) shows that RG is sensibly better than ED and ROS in master utility for most ($WB_\mathcal{A}, p_\mathcal{C}$) combinations. Yet, for follower-worker utility (Figure 1.(c)), RG is almost as good as ED, which is slightly better because a follower worker in ED never cheats. Both, RG and ED, are significantly better than ROS where the deviator $p_\mathcal{C}$ becomes bigger.

The intuition on why this performance is achieved by RG can be obtained from Figure 1.(d), where it can be seen that, for these specific scenarios, our mechanism detects deviations very fast, in contrast with the slow convergence of ED. It should be noticed that fast detection of deviations does not necessarily imply correctness, given that in RG the equilibrium is some $p_\mathcal{C} > 0$. Yet, RG achieves correctness similar or better than ED, and much better than ROS, even though in the latter two the compliant (stable for ED) worker behavior is $p_\mathcal{C} = 0$.
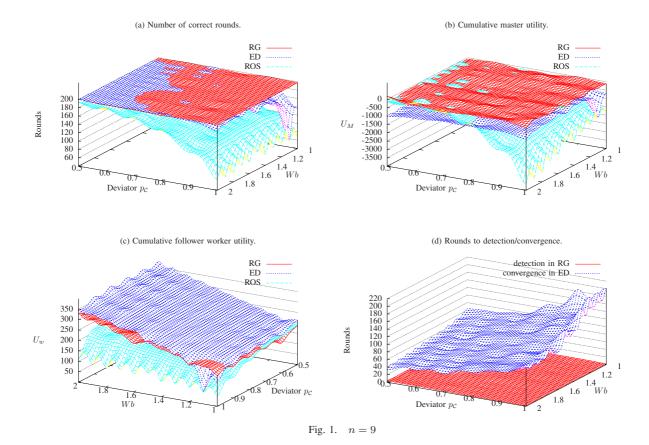
VI. CONCLUSIONS

Our simulations show that in presence of $\lceil n/2 \rceil$ deviators, which is minimal to have an impact on voting schemes, even though the follower workers use $p_\mathcal{C} > 0$ (in contrast with the other mechanisms where $p_\mathcal{C} = 0$ for the followers), even assuming that the cost of verifying is the same as the cost of auditing, and even under the risk of unfair peer-punishment (i.e. workers may be punished even if they do not deviate, because the punishment decision is stochastic), our mechanism performs similarly or better than [13], and both significantly better than a repeated application of [20].

These experimental results, together with our theoretical analysis validating the application of the repeated games framework, demonstrate the benefit and the promise of applying repeated games to the master-worker paradigm. To the best of our knowledge, this is the first study of multi-round master-worker computing applying this framework.

A future extension of this work would be to enable the mechanism to also cope with malicious workers, that is, workers that either intentionally or due to software or hardware errors, return an incorrect task result (recall the relative discussion in the related section of the Introduction). Following [14], we could use statistical information on the distribution of the different worker types (malicious and rational). Then, the deviation threshold of our mechanism will need to be dependent on the expected number of malicious workers, so to keep motivating the rational workers to be truthful; we expect that the analysis will need to be significantly revised. Another future extension would be, as in [20], to consider the possibility of groups of workers colluding in an attempt to increase their utility. For example, one worker could compute the task and inform the others of the correct result so that all return this result to the master (and hence all would obtain the master's payment); or workers would return the same incorrect result in an attempt to cheat the master in accepting an incorrect task result. The challenge here is for the master to cope with these collusions, without knowing which specific workers are colluding.

REFERENCES

[1] "Top500 supercomputing sites," http://www.top500.org/.
[2] E. Korpela, D. Werthimer, D. Anderson, J. Cobb, M. Lebofsky, "SETI@HOME: Massively distributed computing for SETI," *Computing in Science & Engin.*, 3(1):78–83, 2001.
[3] D. P. Anderson, "BOINC: A system for public-resource computing and storage," in *Proc. of the 5th IEEE/ACM International Workshop on Grid Computing,* pp. 4–10, 2004.
[4] Amazon.com, "Amazon's Mechanical Turk," https://www.mturk.com/mturk/welcome.
[5] E. M. Heien, D. P. Anderson, and K. Hagihara, "Computing low latency batches with unreliable workers in volunteer computing environments," *J. of Grid Computing*, 7(4):501–518, 2009.
[6] D. P. Anderson, "Volunteer computing: the ultimate cloud." *ACM Crossroads*, 16(3):7–10, 2010.
[7] D. Kondo, F. Araujo, P. Malecot, P. Domingues, L. Silva, G. Fedak, F. Cappello, "Characterizing result errors in Internet desktop grids," in *Proc. of EUROPAR 2007,* pp. 361–371.
[8] L. F. Sarmenta, "Sabotage-tolerance mechanisms for volunteer computing systems," *Future Generation Computer Systems*, 18(4):561–572, 2002.
[9] K. M. Konwar, S. Rajasekaran, and A. A. Shvartsman, "Robust network supercomputing with malicious processes," in *Proc. of DISC 2006,* pp. 474–488.
[10] A. Fernández Anta, C. Georgiou, L. López, and A. Santos, "Reliable Internet-based master-worker computing in the presence of malicious workers," *Parallel Processing Letters*, 22(1), 2012.
[11] M. Yurkewych, B. N. Levine, and A. L. Rosenberg, "On the cost-ineffectiveness of redundancy in commercial P2P computing," in *Proc. of CCS 2005,* pp. 280–288.
[12] A. Fernández Anta, C. Georgiou, and M. A. Mosteiro, "Designing mechanisms for reliable internet-based computing," in *Proc. of NCA 2008,* pp. 315–324.
[13] E. Christoforou, A. Fernández Anta, C. Georgiou, M. A. Mosteiro, and A. Sánchez, "Applying the dynamics of evolution to achieve reliability in master-worker computing," *Concurrency and Computation: Practice and Experience*, 25(17):2363–2380, 2013.
[14] E. Christoforou, A. Fernández Anta, C. Georgiou, and M. A. Mosteiro, "Algorithmic mechanisms for reliable master-worker Internet-based computing," *IEEE Trans. Computers*, 63(1):179–195, 2014.

(a) Number of correct rounds.

(b) Cumulative master utility.

(c) Cumulative follower worker utility.

(d) Rounds to detection/convergence.

Fig. 1. $n = 9$

[15] S. Choi and R. Buyya, "Group-based adaptive result certification mechanism in Desktop Grids", *Future Generation Comp. Syst.*, 26(5): 776–786, 2010.

[16] A. Tarable, A. Nordio, E. Leonardi, and M. A. Marsan, "The importance of being earnest in crowdsourcing systems", in *Proc. of INFOCOM 2015*, 2821–2829, 2015.

[17] I. Abraham, D. Dolev, R. Gonen, and J. Halpern, "Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation," in *Proc. of PODC 2006*, pp. 53–62.

[18] P. Golle and I. Mironov, "Uncheatable distributed computations," in *Topics in Cryptology-CT–RSA 2001*, pp. 425–440.

[19] J. Shneidman and D. C. Parkes, "Rationality and self-interest in peer to peer networks," in *Peer-to-Peer Systems II*, pp. 139–148, 2003.

[20] A. Fernández Anta, C. Georgiou, M. A. Mosteiro, and D. Pareja, "Algorithmic mechanisms for reliable crowdsourcing computation under collusion," *PLoS ONE*, 10(3):e0116520, 2015.

[21] J. M. Smith, *Evolution and the Theory of Games*. Cambridge University Press, 1982.

[22] C. Szepesvári, "Algorithms for reinforcement learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 4(1):1–103, 2010.

[23] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*. The MIT Press, 1994.

[24] E. Christoforou, A. Fernández Anta, C. Georgiou, M. A. Mosteiro, and A. Sánchez, "Reputation-based mechanisms for evolutionary master-worker computing," in *Proc. of OPODIS 2013*, pp. 98–113.

[25] J. Chen, S. McCauley, and S. Singh, "Rational Proofs with Multiple Provers", in *Proc. of the 7th ACM Conference on Innovations in Theoretical Computer Science*, pp. 237–248, 2016.

[26] M. Taufer, D. P. Anderson, P. Cicotti, and C. L. Brooks III, "Homogeneous redundancy: a technique to ensure integrity of molecular simulation results using public computing," in *Proc. of IPDPS 2005*.

[27] P. Golle and S. Stubblebine, "Secure distributed computing in a commercial environment," *Financial Cryptography*, 2001.

[28] D. Lázaro, D. Kondo, and J. M. Marqués, "Long-term availability prediction for groups of volunteer resources," *Journal of Parallel and Distributed Computing*, 72(2):281–296, 2012.

[29] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[30] A. Fernández Anta, C. Georgiou, M. A. Mosteiro, D. Pareja, "Multi-round Master-Worker Computing: a Repeated Game Approach", *CoRR*, http://arxiv.org/abs/1508.05769, 2015.

APPENDIX

A. Proof of Lemma 4

Proof: Let σ be a mixed strategy profile, that is, a mapping from workers to probability distributions over the pure strategies $\{\mathcal{C}, \overline{\mathcal{C}}\}$, let σ_i be the probability distribution $\{p_{\mathcal{C}_i}, (1 - p_{\mathcal{C}_i})\}$ of worker i in σ, and let σ_{-i} be the mixed strategy profile of all workers but i. We want to find worker i's minmax payoff, which is the lowest payoff that the other workers can force upon i (cf. [23]). That is,

$$v_i = \min_{\sigma_{-i}} \max_{\sigma_i} U_i(\sigma_i, \sigma_{-i}).$$

For any worker i, there are four possible utility outcomes, namely.

$$U_i(s_i = \mathcal{C}, |F| > n/2) = (1 - p_{\mathcal{V}}) WB_{\mathcal{A}} - p_{\mathcal{V}} WP_{\mathcal{C}}|F|$$
$$U_i(s_i = \overline{\mathcal{C}}, |F| > n/2) = p_{\mathcal{V}} WB_{\mathcal{A}} - WC_{\mathcal{T}}$$
$$U_i(s_i = \mathcal{C}, |F| < n/2) = -p_{\mathcal{V}} WP_{\mathcal{C}}|F|$$
$$U_i(s_i = \overline{\mathcal{C}}, |F| < n/2) = WB_{\mathcal{A}} - WC_{\mathcal{T}}.$$

The expected utility of a worker i that deviates from equilibrium when all other workers use the same mixed strategy $p_{\mathcal{C}_{-i}}$ is the following.

$$U_i = p_{\mathcal{C}_i}\left(Pr\left(|F_{-i}| \geq \frac{n-1}{2}\right)U_i\left(s_i = \mathcal{C}, |F| > \frac{n}{2}\right)\right.$$
$$+ Pr\left(|F_{-i}| < \frac{n-1}{2}\right)U_i\left(s_i = \mathcal{C}, |F| < \frac{n}{2}\right))$$
$$+ (1 - p_{\mathcal{C}_i})(Pr\left(|F_{-i}| > \frac{n-1}{2}\right)U_i\left(s_i = \overline{\mathcal{C}}, |F| > \frac{n}{2}\right)$$
$$+ Pr\left(|F_{-i}| \leq \frac{n-1}{2}\right)U_i\left(s_i = \overline{\mathcal{C}}, |F| < \frac{n}{2}\right)).$$

It can be seen that U_i is linear with respect to $p_{\mathcal{C}_i}$. That is, the function is either monotonically increasing, monotonically decreasing, or constant with respect to $p_{\mathcal{C}_i}$, depending on the relation among the parameters, but it does not have critical points. Hence, for any given $p_{\mathcal{C}_{-i}}$ the maximum utility for worker i occurs when $p_{\mathcal{C}_i}$ is either 0 or 1. We get then that the maximum U_i is either

$$U_i(p_{\mathcal{C}_i} = 1) = Pr\left(|F_{-i}| \geq \frac{n-1}{2}\right)U_i\left(s_i = \mathcal{C}, |F| > \frac{n}{2}\right)$$
$$+ Pr\left(|F_{-i}| < \frac{n-1}{2}\right)U_i\left(s_i = \mathcal{C}, |F| < \frac{n}{2}\right), \text{ or}$$
$$U_i(p_{\mathcal{C}_i} = 0) = Pr\left(|F_{-i}| > \frac{n-1}{2}\right)U_i\left(s_i = \overline{\mathcal{C}}, |F| > \frac{n}{2}\right)$$
$$+ Pr\left(|F_{-i}| \leq \frac{n-1}{2}\right)U_i\left(s_i = \overline{\mathcal{C}}, |F| < \frac{n}{2}\right).$$

Replacing and using that $\sum_{j=0}^{n-1}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j} = 1$, and that $\sum_{j=0}^{n-1}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}j = (n-1)p_{\mathcal{C}_{-i}}$ we have the following.

$$U_i(p_{\mathcal{C}_i} = 1) = \sum_{j=(n-1)/2}^{n-1}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}$$
$$\cdot((1 - p_{\mathcal{V}})WB_{\mathcal{A}} - p_{\mathcal{V}}WP_{\mathcal{C}}(j+1))$$
$$+ \sum_{j=0}^{(n-3)/2}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}(-p_{\mathcal{V}}WP_{\mathcal{C}}(j+1))$$
$$= -p_{\mathcal{V}}WP_{\mathcal{C}}(1 + (n-1)p_{\mathcal{C}_{-i}})$$
$$+ (1 - p_{\mathcal{V}})WB_{\mathcal{A}}\sum_{j=(n-1)/2}^{n-1}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}.$$
$$(9)$$

And

$$U_i(p_{\mathcal{C}_i} = 0)$$
$$= \sum_{j=(n+1)/2}^{n-1}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}(p_{\mathcal{V}}WB_{\mathcal{A}} - WC_{\mathcal{T}})$$
$$+ \sum_{j=0}^{(n-1)/2}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}(WB_{\mathcal{A}} - WC_{\mathcal{T}})$$
$$= p_{\mathcal{V}}WB_{\mathcal{A}} - WC_{\mathcal{T}}$$
$$+ (1 - p_{\mathcal{V}})WB_{\mathcal{A}}\sum_{j=0}^{(n-1)/2}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}.$$
$$(10)$$

To find the minmax payoff, we want to find the mixed strategy (i.e., $p_{\mathcal{C}_{-i}}$) that other workers may choose to minimize these utilities. Equation 10 is minimized when $p_{\mathcal{C}_{-i}} = 1$, yielding $U_i(p_{\mathcal{C}_i} = 0, p_{\mathcal{C}_{-i}} = 1) = p_{\mathcal{V}}WB_{\mathcal{A}} - WC_{\mathcal{T}} \geq 0$. The latter inequality is true because $p_{\mathcal{V}} \geq WC_{\mathcal{T}}/WB_{\mathcal{A}}$. On the other hand, given that $p_{\mathcal{V}} > 2WB_{\mathcal{A}}/(2WB_{\mathcal{A}} + WP_{\mathcal{C}}n)$, it is $p_{\mathcal{V}}WP_{\mathcal{C}}n > 2(1 - p_{\mathcal{V}})WB_{\mathcal{A}}$. Then, from Equation 9, we have

$$U_i(p_{\mathcal{C}_i} = 1) < -p_{\mathcal{V}}WP_{\mathcal{C}}(1 - p_{\mathcal{C}_{-i}}) - 2(1 - p_{\mathcal{V}})WB_{\mathcal{A}}p_{\mathcal{C}_{-i}}$$
$$+ (1 - p_{\mathcal{V}})WB_{\mathcal{A}}\sum_{j=(n-1)/2}^{n-1}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}$$
$$= -p_{\mathcal{V}}WP_{\mathcal{C}}(1 - p_{\mathcal{C}_{-i}}) + (1 - p_{\mathcal{V}})WB_{\mathcal{A}}$$
$$\cdot\left(-2p_{\mathcal{C}_{-i}} + \sum_{j=(n-1)/2}^{n-1}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1-p_{\mathcal{C}_{-i}})^{n-1-j}\right).$$
$$(11)$$

Given that $\left(-2p_{\mathcal{C}_{-i}} + \sum_{j=(n-1)/2}^{n-1}\binom{n-1}{j}p_{\mathcal{C}_{-i}}^j(1 - p_{\mathcal{C}_{-i}})^{n-1-j}\right) \leq 0$, Equation 11 is negative for any $p_{\mathcal{C}_{-i}}$. Therefore, $U_i(p_{\mathcal{C}_i} = 0, p_{\mathcal{C}_{-i}} = 1) \geq U_i(p_{\mathcal{C}_i} = 1, p_{\mathcal{C}_{-i}} = p)$ for any $p \in [0, 1]$, and the minmax expected payoff is $v_i = U_i(p_{\mathcal{C}_i} = 0, p_{\mathcal{C}_{-i}} = 1) = p_{\mathcal{V}}WB_{\mathcal{A}} - WC_{\mathcal{T}}$. ∎