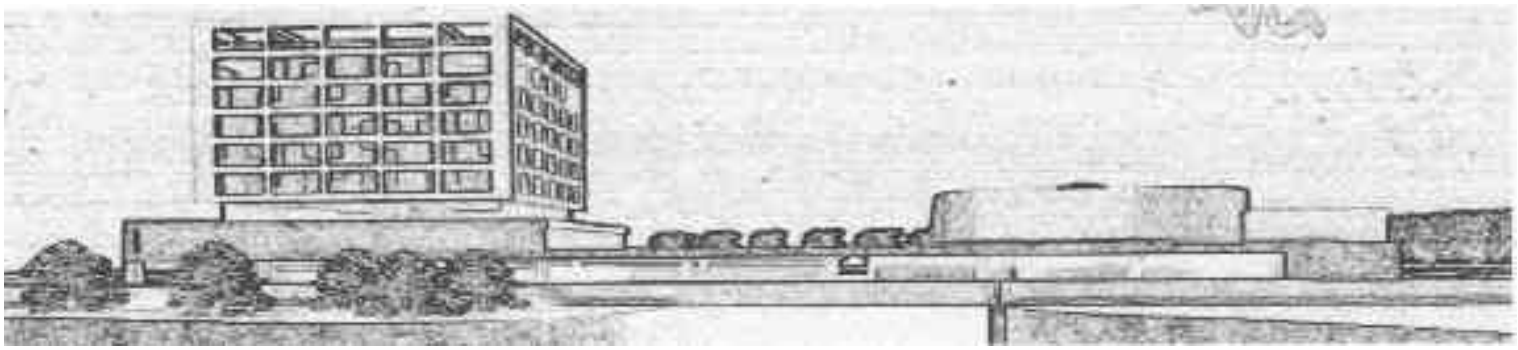
VOLUME IX, NUMBER 3



REPORTS ON SYSTEMS AND COMMUNICATIONS

Móstoles (Madrid), September 2009
Depósito Legal: M-50653-2004
ISSN: 1698-7489

Table of Contents

Algorithmic Mechanisms for Internet-based Master-Worker Computing with Untrusted and Selfish Workers *

Antonio Fernández Anta[1], Chryssis Georgiou[2], and Miguel A. Mosteiro[1,3]

LADyR, GSyC, Universidad Rey Juan Carlos
anto@gsyc.es, miguel.mosteiro@urjc.es
Department of Computer Science, University of Cyprus
chryssis@cs.ucy.ac.cy
Department of Computer Science, Rutgers University
mosteiro@cs.rutgers.edu

Abstract. We consider Internet-based master-worker computations, where a master processor assigns, across the Internet, a computational task to a set of untrusted worker processors, and collects their responses; examples of such computations are the "@home" projects such as SETI. Prior work dealing with Internet-based task computations has either considered only rational, or only malicious and altruistic workers. Altruistic workers always return the correct result of the task, malicious workers always return an incorrect result, and rational workers act based on their self-interest. However, in a massive computation platform, such as the Internet, it is expected that all three type of workers coexist. Therefore, in this work we study Internet-based master-worker computations in the presence of Malicious, Altruistic, and Rational workers. A stochastic distribution of the workers over the three types is assumed. Considering all the three types of workers renders a combination of game-theoretic and classical distributed computing approaches to the design of mechanisms for reliable Internet-based computing. Indeed, in this work such an algorithmic mechanism that makes use of realistic incentives to obtain the correct task result with a parametrized probability is designed. Only when necessary, the incentives are used to force the rational players to a certain equilibrium (which forces the workers to be truthful) that overcomes the attempt of the malicious workers to deceive the master. Finally, the mechanism is analyzed in two realistic Internet-based master-worker applications. This work is an example of how game theory can be used as a tool to formalize and solve a practical Distributed Computing problem such as Internet supercomputing.

Keywords: Internet-based computing, Algorithmic mechanism, Task execution, Rational, altruistic, and malicious processors, Volunteering computing.

* This research was supported in part by Comunidad de Madrid grant S-0505/TIC/0285; Spanish MICINN grant TIN2008-06735-C02-01; Spanish MEC grant TIN2005-09198-C02-01; EU IST #15964 (AEOLUS); EU Marie Curie European Reintegration Grant IRG 210021; and NSF grant CCF 0632838.

1 Introduction

Motivation and Prior Work. In recent years, the Internet has become an alternative (to expensive supercomputing parallel machines) computational platform for processing complex computational jobs. Examples of such Internet-based processing are the "@home" projects [3], such as SETI [23] (a classical example of *volunteer computing*) and Grid computing [8]. However, Internet-based computing (referred sometimes as P2P computing–P2PC [13, 39]) has not reached its full potential due to the untrustworthiness nature of the platform's components [3, 16]. Typically, in Internet-based computing (e.g, in SETI) the following Master-Worker approach is employed: A master computer sends jobs (or *tasks*), across the Internet, to worker computers that are willing to execute them. These workers execute and report back the result of the task computation. However, these workers are unavoidably non-trustworthy, and hence might report incorrect results. Naturally, the master attempts to minimize the impact of these bogus results (and increase its chance of obtaining the correct task result) by assigning the same task to several workers and comparing their outcomes (that is, *redundant* task allocation is employed [3]).

This problem has recently been studied under two different views: from a "classical" distributed computing view [11, 22] and from a game-theoretic view [12, 39]. Under the first view, the workers are classified as either *malicious* (Byzantine) or *altruistic*, based on a predefined behavior. The malicious workers have a "bad" behavior which results in reporting an incorrect result to the master. This behavior is, for example, due to a hardware or a software error or due to an ill-state of the worker (it behaves maliciously intentionally). Altruistic workers exhibit a "good" behavior, that is, they compute and truthfully return the correct task result (they are essentially the "correct" nodes). Under this view, "classical" distributed computing models are defined (e.g., a fixed bound on the probability of malicious nodes is assumed) and typical malicious-tolerant voting protocols are designed.

Under the game-theoretic view, workers act on their own *self-interest* and they do not have an a priori established behavior, that is, they are assumed to be *rational* [1, 16]. In other words, the workers decide on whether they will be *honest* (and hence compute and truthfully report the correct task result) or *cheat* (and hence report a bogus result) depending on which strategy increases their benefit (utility). Under this view, Algorithmic Mechanisms [1, 7, 32] are employed, where games are designed to provide the necessary incentives so that processors' interests are best served by acting "correctly." In particular, the master provides some reward (resp. penalty) should a worker be honest (resp. cheat). The design objective is for the master to force a desired unique *Nash equilibrium* (NE) [31], i.e., a strategy choice by each worker such that none of them has incentive to change it. That Nash equilibrium is the one in which the master achieves a desired probability of obtaining the correct task result.

The above views could complement one another, if a certain computation includes only malicious and altruistic workers, or only rational workers. However, the pragmatic situation on the Internet is different: all three type of workers might co-exist in a given computation. One could assume that all workers are rational, but what, for example, if a software bug occurs that makes a worker deviate from its protocol, and hence compute and return an incorrect result? This worker is no longer exhibiting a rational behavior,

but rather an erroneous or irrational one (that from the master's point of view it can be seen as malicious).

Contributions. In this work we consider Internet-based master-worker computations where Malicious, Altruistic and Rational workers co-exist. To the best of our knowledge, this is the first work that considers such co-existence in Internet-based master-worker (P2PC) computing. Considering all the three types of workers renders a combination of game-theoretic and classical distributed computing approaches to the design of mechanisms for reliable Internet-based computing. In particular

- A collection of realistic payoff parameters and reward models are identified (Section 2) and the above Internet-based master-worker computation problem is formalized as a *Bayesian game* [20] (Section 3). There is a probability distribution of workers among the worker types. The master and the workers do not know the type of other workers, only the probability distribution. The rational workers play a game looking for a Nash Equilibrium, while the malicious and altruistic workers have a predefined strategy to cheat or be honest, respectively. The master does not participate in the game, it designs the game to be played.
- We design a general voting algorithm that the master runs to implement the above-mentioned game (Section 3). The algorithm is parametrized in terms of a probability of auditing $p_{\mathcal{A}}$ (defined in Section 3). Under a general type probability distribution, we analyze the master's utility and probability of error (probability of obtaining the incorrect task result) and identify the conditions under which the game has Nash Equilibria.
- Based on specific type probability distributions, an algorithmic mechanism in which the master chooses the values of $p_{\mathcal{A}}$ to guarantee a parametrized bound on the probability of error is designed (Section 4). Once this is achieved, the master also attempts to maximize its utility. Note that the mechanism designed (and its analysis) is general in that reward models can either be fixed exogenously or be chosen by the master. It is also shown that this mechanism is the only feasible approach for the master to achieve a given bound on the probability of error.
- Finally, under the constrain of the bounded probability of error, it is shown how to maximize the master utility in two realistic scenarios (Section 5). The first scenario abstracts a system of volunteering computing like SETI, and the second, a company that buys computing cycles from Internet computers and sells them to its customers in the form of a task-computation service.

Related work. Prior examples of game theory in distributed computing include work on Internet routing [15, 24, 28, 35], resource/facility location and sharing [14, 17], containment of viruses spreading [30], secret sharing [1, 19], P2P services [2, 25, 26] and task computations [12, 39]. For more discussion on the connection between game theory and computing we refer the reader to the survey by Halpern [18] and the book by Nisan et al [33].

Distributed computation in presence of selfishness was also studied within the scope of *Combinatorial Agencies* in Economics [4]. The computation is carried out as a game of complete information where only rational players are considered. The goal in that

work is to study how the utility of the master is affected if the equilibria space is limited to pure strategies. To that extent, the computation of a few Boolean functions is evaluated. If the parameters of the problem yield multiple mixed equilibrium points, it is assumed that workers accept one "suggested" by the master.

Eliaz [9] seems to be the first to formally study the co-existence of Byzantine (malicious) and rational players. He introduces the notion of *k-fault-tolerant Nash Equilibrium* as a state in which no player benefits from unilaterally deviating despite up to k players acting maliciously. He demonstrates this concept by designing simple mechanisms that implement the constrained Walrasian function and a choice rule for the efficient allocation of an indivisible good (e.g., in auctions). Abraham et al [1] extend Eliaz's concept to accommodate colluding rational players. In particular they design a secret sharing protocol and prove that it is (k, t)-robust, that is, it is correct despite up to k colluding rational players and t Byzantine ones.

Aiyer et al. [2] introduce the BAR model to reason about systems with Byzantine (malicious), Altruistic, and Rational participants. They also introduce the notion of a protocol being BAR-tolerant, that is, the protocol is resilient to both Byzantine faults and rational manipulation. (With this respect, one might say that our algorithmic mechanism designed in this work is BAR-tolerant.) As an application, they designed a cooperative backup service for P2P systems, based on a BAR-tolerant replicated state machine. Li et al [26] also considered the BAR model to design a P2P live streaming application based on a BAR-tolerant gossip protocol. Both works employ incentive-based game theoretic techniques (to remove the selfish behavior), but the emphasis is on building a reasonably practical system (hence, formal analysis is traded for practicality). Recently, Li et al [25] developed a P2P streaming application, called FlightPath, that provides a highly reliable data stream to a dynamic set of peers. FlightPath, as opposed to the abovementioned BAR-based works, is based on mechanisms for *approximate equilibria* [6], rather than strict equilibria. In particular, ϵ-Nash equilibria are considered, in which rational players deviate only if and only if they expect to benefit by more than a factor of ϵ. As the authors claim, the less restrictive nature of these equilibria enables the design of incentives to limit selfish behavior rigorously, while it provides sufficient flexibility to build practical systems.

Very recently, Gairing [15], introduced and studied *malicious Bayesian congestion games*. These games extend congestion games [36] by allowing players to act in a malicious way. In particular, each player can either be rational or, with a certain probability, be malicious (with the sole goal of disturbing the other players). As in our work, players are not aware of each other's type, and this uncertainty is described by a probability distribution. Among other results, Gairing shows that, unlike congestion games, these games do not in general possess a Nash Equilibrium in pure strategies. Also he studies the impact of malicious types on the social cost (the overall performance of the system) by measuring the so-called *Price of Malice*. This measure was first introduced by Moscibroda et al [30] to measure the influence of malicious behavior for a virus inoculation game involving both rational (selfish) and malicious nodes.

2 Definitions and Notation

System model. The assumed distributed system is formed by a master processor M and a set W of $n = |W|$ workers. We assume that the master chooses n to be odd. The master has a task that wants to compute. For some reason, the master does not compute the task itself, but chooses to send it to *all* the workers, wait for their answers, and decide on a value that it believes to be the correct output of the task. The tasks considered in this work are assumed to have a unique solution.

Each of the n workers has one of the following types, *rational*, *malicious*, or *altruistic*. The exact number of workers of each type is unknown. However, it is known that each worker is independently of one of the three types with probabilities p_ρ, p_μ, p_α, respectively, where $p_\rho + p_\mu + p_\alpha = 1$. Malicious and altruistic workers always cheat and are honest, respectively, independently of how such a behavior impacts their utilities. In the context of this paper, being honest means returning the correct value, and cheating means returning some incorrect value. On the other hand, rational workers are assumed to be selfish in a game-theoretic sense, i.e., their aim is to maximize their benefit (utility) under the assumption that other workers do the same. Hence, they will be honest or cheat depending on which strategy maximizes their utility. While it is assumed that rational players make their decision individually, it is assumed that all the (malicious and rational) workers that cheat return the same incorrect value. This yields a worst case scenario (and hence analysis) for the master with respect to its probability of obtaining the correct result. (In some sense, this can be seen as a cost-free, weak form of collusion). Finally, it is assumed that all workers reply (abstention is not allowed) and that all their answers reach the master.

In order to model the individuality of the non-monetary part of each rational worker benefit/penalty, the distribution over types could be generalized to different types of rational workers instead of one. More precisely, define a probability distribution over each possible combination of payoffs in $\mathbb{R}^4$, restricting signs appropriately, so that each rational worker draws independently its strategic normal form from this distribution. However, the analysis presented here would be the same but using expected payoffs, the expectation taken over such distribution. Thus, for the sake of clarity and without loss of generality, we assume that the strategic normal form is unique for all players, i.e., all rational workers are of the same type.

The objective of the master is twofold. First, the master has to guarantee that the decided value is correct with probability at least $1 - \varepsilon$, for a known constant $0 \leq \varepsilon < 1$. Then, having achieved this, the master wants to maximize its own benefit (utility). To achieve this it has two weapons. On the one hand, it can audit the response of the workers (at a cost). In particular, the master computes the task by itself, and checks which workers have been truthful or not. (From the assumptions that cheaters return the same incorrect answer and tasks have unique solutions, it follows that there can only be two kind of replies – a correct and an incorrect one.) On the other hand, the master can punish and reward workers, which can be used (possibly combined with audit) to encourage rational workers to be honest. When the master audits, it can accurately punish and reward workers. However, when the replies are not audited, rewards and penalties can be applied following different models.

The reward models considered in this paper are presented in Table 1. Two of the models reward or penalize a worker depending on whether its reply is equal to the majority of replies (observe that at most two replies are possible, and since n is odd, one reply has majority). These reward models are sensible when the probability of a majority of honest replies is reasonably large. Observe as well that three models do not punish (some even reward) the workers whose reply is in the minority. This tries to avoid punishing honest workers that are outnumbered by cheaters. The payoff parameters used are detailed in Table 2. All these parameters are non-negative. Observe that there are different parameters for the reward $WB_{\mathcal{Y}}$ to a worker and the cost $MC_{\mathcal{Y}}$ 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. In fact, in some applications they may be completely unrelated, as for example in the SETI-like scenario presented in Section 5.1. It is assumed that $WB_{\mathcal{Y}}$ and $WP_{\mathcal{C}}$ are chosen by the master whereas the other payoff parameters and the reward models can be fixed exogenously.

$\mathcal{R}_{\pm}$	the master rewards the majority and penalizes the minority
$\mathcal{R}_{\mathrm{m}}$	the master rewards the majority only
$\mathcal{R}_{\mathrm{a}}$	the master rewards all workers
$\mathcal{R}_{\emptyset}$	the master does not reward any worker

Table 1. Reward models

Game Theory concepts. We study the problem under the assumption that the rational workers, or *players*, will play a game looking for an equilibrium (malicious and altruistic workers have a predefined strategy to cheat or be honest, respectively). The master does not play the game, it only defines the protocol and the parameters to be followed (i.e., it designs the game or mechanism). The master and the workers do not know the type of other workers, only the probability distribution. Hence, the game played is a so-called game with imperfect information or *Bayesian game* [20]. The action space is the set of pure strategies $\{\mathcal{C}, \overline{\mathcal{C}}\}$, and the belief of a player is the probability distribution over types. Each player knows in advance the distribution over types, the total number of workers, and its normal strategic form, which is assumed to be unique. The game formulation is given in the next section.

$WP_{\mathcal{C}}$	worker's punishment for being caught cheating
$WC_{\mathcal{T}}$	worker's cost for computing the task
$WB_{\mathcal{Y}}$	worker's benefit from master's acceptance
$MP_{\mathcal{W}}$	master's punishment for accepting a wrong answer
$MC_{\mathcal{Y}}$	master's cost for accepting the worker's answer
$MC_{\mathcal{A}}$	master's cost for auditing worker's answers
$MB_{\mathcal{R}}$	master's benefit from accepting the right answer

Table 2. Payoffs

Recall from [34], that for any finite game, a mixed strategy profile σ is a *mixed-strategy Nash equilibrium* (MSNE) if, and only if, for each player i,

$$U_i(s_i, \sigma_{-i}) = U_i(s_i', \sigma_{-i}), \forall s_i, s_i' \in supp(\sigma_i),$$
$$U_i(s_i, \sigma_{-i}) \geq U_i(s_i', \sigma_{-i}), \forall s_i, s_i' : s_i \in supp(\sigma_i), s_i' \notin supp(\sigma_i),$$

where s_i is the strategy used by player i in the strategy profile s, σ_i is the probability distribution over pure strategies used by player i in σ, σ_{-i} is the probability distribution over pure strategies used by each player but i in σ, $U_i(s_i, \sigma_{-i})$ is the expected utility of player i when using strategy s_i with mixed strategy profile σ, and $supp(\sigma_i)$ is the set of strategies in σ with positive probability.

In words, given a MSNE with mixed-strategy profile σ, for each player i, the expected utility, assuming that all other players do not change their choice, is the same for each pure strategy that the player can choose with positive probability in σ, and it is not less than the expected utility of any pure strategy with probability zero of being chosen in σ. Then, in order to find conditions for equilibria, we want to study for each player i

$$\Delta U_i \triangleq U_i(s_i = \mathcal{C}, \sigma_{-i}) - U_i(s_i = \overline{\mathcal{C}}, \sigma_{-i}).$$

If we show conditions such that $\Delta U = 0$, then we have a MSNE.[1] If we denote by $p_{\mathcal{C}}$ the probability that player i cheats, then in the MSNE $0 \neq p_{\mathcal{C}} \neq 1$. On the other hand, if we show conditions that make $\Delta U < 0$ for each player i, we know that there is a pure strategies NE where all players choose to be honest, i.e. $p_{\mathcal{C}} = 0$. (There is no NE where some players choose a pure strategy and others do not because the game is symmetric for all rational players. If a distribution over many types of rational players is defined, then we would have to consider such a NE.)

The following notation will be used throughout.

$$\mathbf{P}_q^{(n)}(a, b) \triangleq \sum_{i=a}^{b} \binom{n}{i} q^i (1-q)^{n-i}$$

A table summarizing the notation used throughout the paper is given in the Appendix.

3 Game Definition and Analysis

In this section we present the protocol that the master uses to obtain the result of the task. The protocol essentially implements the game to be played by the (rational) workers, which we also define in this section. Finally we analyze the game under a general type probability distribution.

[1] Given that the utility is the same for all players, we refer to ΔU_i as ΔU.

3.1 Protocol Description

The basic protocol used by the master to accept the task result can be described as follows. After receiving the replies from all workers, and independently of the distribution of the answers, the master processor chooses to audit the answers with some probability $p_{\mathcal{A}}$. If the answers were not audited it accepts the result of the majority. Then, it applies the corresponding reward model. The protocol is detailed in Algorithm 1. The specific values of $p_{\mathcal{A}}$ are chosen in the next sections according with the known type distribution of workers and payoffs.

For computational reasons, besides $p_{\mathcal{A}}$ and the task to be computed, the master also sends a certificate. The certificate includes the strategy that if the rational workers play will lead them to the unique NE, together with the appropriate data to demonstrate this fact. More details for the use of the certificate are given in Section 4.5.

Algorithm 1: Master algorithm

```
1 send (task, p_A, certificate) to all the workers in W
2 upon receiving all answers do
3     audit the answers with probability p_A
4     if the answers were not audited then accept the majority
5     apply the reward model
```

As discussed in Section 2, there are only two values returned to the master – the correct value and a unique incorrect one. Together with the fact that the master chooses n to be odd, in line 4 it is not possible to have relative majority. Considering relative majority could be made possible by making appropriate changes to the model and to the mechanism analysis. However, the analysis becomes more involved while not giving more insight to the problem under study.

3.2 Game Definition

Putting together the game-related discussion in Section 2 and the above protocol, we formulate the Internet-based Master Worker computation considered in this works as the following Bayesian game

$$\mathcal{G}(W, \varepsilon, \mathcal{D}, A, p_{\mathcal{A}}, \mathcal{R}, pfs),$$

where W is the set of n workers, $0 \leq \varepsilon < 1$ is the error probability, $\mathcal{D}$ is the type probability distribution $(p_\rho, p_\mu, p_\alpha)$, $A = \{\mathcal{C}, \overline{\mathcal{C}}\}$ is the workers' actions space (recall that only rational players have a probabilistic choice over pure strategies, malicious workers always cheat and altruistic workers are always honest), $p_{\mathcal{A}}$ is as described in Algorithm 1, $\mathcal{R}$ is one of the reward models given in Table 1, and pfs are the payoffs as described in Table 2. Recall that the master and the workers do not know the other workers types, but $\mathcal{D}$ is known.

As mentioned before, the master does not participate in the game, but it designs the game to be played. In particular, the master runs Algorithm 1 after using a mechanism designed in Section 4. In order to obtain a mechanism that is useful for any scenario we do not restrict ourselves to a particular instance of payoffs or reward models. Instead, we leave those variables as parameters and focus our study on how to choose $p_{\mathcal{A}}$ to have the probability of error bounded by ε. Were payoffs and reward models a choice of the master, its utility can be maximized choosing those parameters conveniently in Equation 2 (given below). Two realistic examples are given in Section 5.

3.3 Game Analysis

We now analyze the game under a general type probability distribution. In the next section we design a mechanism for specific families of type probability distributions.

Error Probability and Master Utility. Recall that n is assumed to be odd. Letting $q = p_\mu + p_\rho p_{\mathcal{C}}$, where $p_{\mathcal{C}}$ is the probability that a rational player chooses strategy $\mathcal{C}$, the probability that the master obtains the wrong answer is

$$P_{wrong} = (1 - p_{\mathcal{A}})\mathbf{P}_q^{(n)}(\lceil n/2 \rceil, n). \tag{1}$$

On the other hand, the expected utility of the master is

$$\begin{aligned} U_M = p_{\mathcal{A}}\big(&MB_{\mathcal{R}} - MC_{\mathcal{A}} - n(1-q)MC_{\mathcal{Y}}\big) \\ &+ (1 - p_{\mathcal{A}})\big(MB_{\mathcal{R}}\mathbf{P}_q^{(n)}(0, \lfloor n/2 \rfloor) - MP_{\mathcal{W}}\mathbf{P}_q^{(n)}(\lceil n/2 \rceil, n) + \gamma\big). \end{aligned} \tag{2}$$

Where,

$$\gamma = \begin{cases} -MC_{\mathcal{Y}}(\mathbf{E}_{1-q}^{(n)}(\lceil n/2 \rceil, n) + \mathbf{E}_q^{(n)}(\lceil n/2 \rceil, n)) & \text{for the } \mathcal{R}_{\mathrm{m}} \text{ and } \mathcal{R}_{\pm} \text{ models.} \\ -nMC_{\mathcal{Y}} & \text{for the } \mathcal{R}_{\mathrm{a}} \text{ model.} \\ 0 & \text{for the } \mathcal{R}_{\emptyset} \text{ model.} \end{cases}$$

and $\mathbf{E}_p^{(n)}(a, b) \triangleq \sum_{i=a}^{b} \binom{n}{i} i p^i (1-p)^{n-i}$, $p \in [0, 1]$.

Equilibria Conditions. For any player i, let $w_{s_i}^{\mathcal{C}}$ be the payoff of player i when using strategy s_i in the strategy profile s if the majority of workers cheat and the master does not audit, $w_{s_i}^{\overline{\mathcal{C}}}$ if the minority of workers cheat and the master does not audit, and $w_{s_i}^{\mathcal{A}}$ otherwise.

Using this notation, the payoffs for each reward model, are detailed in Table 3.

Then, for each player i,

$$\begin{aligned} \Delta U = (w_{\mathcal{C}}^{\mathcal{A}} - w_{\overline{\mathcal{C}}}^{\mathcal{A}})p_{\mathcal{A}} + (1 - p_{\mathcal{A}}) \\ \Bigg((w_{\mathcal{C}}^{\mathcal{C}} - w_{\overline{\mathcal{C}}}^{\mathcal{C}})\mathbf{P}_q^{(n-1)}(\lceil n/2 \rceil, n-1) + (w_{\mathcal{C}}^{\overline{\mathcal{C}}} - w_{\overline{\mathcal{C}}}^{\overline{\mathcal{C}}})\mathbf{P}_q^{(n-1)}(0, \lfloor n/2 \rfloor - 1) \\ + (w_{\mathcal{C}}^{\mathcal{C}} - w_{\overline{\mathcal{C}}}^{\overline{\mathcal{C}}})\binom{n-1}{\lfloor n/2 \rfloor} q^{\lfloor n/2 \rfloor}(1-q)^{\lfloor n/2 \rfloor} \Bigg). \end{aligned}$$

	$\mathcal{R}_{\pm}$	$\mathcal{R}_{\mathrm{m}}$	$\mathcal{R}_{\mathrm{a}}$	$\mathcal{R}_{\emptyset}$
$w_{\mathcal{C}}^{\mathcal{A}}$	$-WP_{\mathcal{C}}$	$-WP_{\mathcal{C}}$	$-WP_{\mathcal{C}}$	$-WP_{\mathcal{C}}$
$w_{\overline{\mathcal{C}}}^{\mathcal{A}}$	$WB_{\mathcal{Y}} - WC_{\mathcal{T}}$	$WB_{\mathcal{Y}} - WC_{\mathcal{T}}$	$WB_{\mathcal{Y}} - WC_{\mathcal{T}}$	$WB_{\mathcal{Y}} - WC_{\mathcal{T}}$
$w_{\mathcal{C}}^{\mathcal{C}}$	$WB_{\mathcal{Y}}$	$WB_{\mathcal{Y}}$	$WB_{\mathcal{Y}}$	0
$w_{\overline{\mathcal{C}}}^{\mathcal{C}}$	$-WP_{\mathcal{C}} - WC_{\mathcal{T}}$	$-WC_{\mathcal{T}}$	$WB_{\mathcal{Y}} - WC_{\mathcal{T}}$	$-WC_{\mathcal{T}}$
$w_{\mathcal{C}}^{\overline{\mathcal{C}}}$	$-WP_{\mathcal{C}}$	0	$WB_{\mathcal{Y}}$	0
$w_{\overline{\mathcal{C}}}^{\overline{\mathcal{C}}}$	$WB_{\mathcal{Y}} - WC_{\mathcal{T}}$	$WB_{\mathcal{Y}} - WC_{\mathcal{T}}$	$WB_{\mathcal{Y}} - WC_{\mathcal{T}}$	$-WC_{\mathcal{T}}$

Table 3. Payoffs for each reward model.

Notice in Table 3 that $w_{\mathcal{C}}^{\mathcal{A}} - w_{\overline{\mathcal{C}}}^{\mathcal{A}} = WC_{\mathcal{T}} - WP_{\mathcal{C}} - WB_{\mathcal{Y}}$ for all models. Also notice from Table 3 that, for any reward model, $w_{\overline{\mathcal{C}}}^{\mathcal{C}} = w_{\mathcal{C}}^{\overline{\mathcal{C}}} - WC_{\mathcal{T}}$ and $w_{\overline{\mathcal{C}}}^{\overline{\mathcal{C}}} = w_{\mathcal{C}}^{\mathcal{C}} - WC_{\mathcal{T}}$. Replacing,

$$\begin{aligned}\Delta U = {} & WC_{\mathcal{T}} - p_{\mathcal{A}}(WP_{\mathcal{C}} + WB_{\mathcal{Y}}) + (1 - p_{\mathcal{A}}) \\ & (w_{\mathcal{C}}^{\mathcal{C}}(\mathbf{P}_q^{(n-1)}(\lfloor n/2 \rfloor, n-1) - \mathbf{P}_q^{(n-1)}(0, \lfloor n/2 \rfloor)) \\ & + w_{\mathcal{C}}^{\overline{\mathcal{C}}}(\mathbf{P}_q^{(n-1)}(0, \lfloor n/2 \rfloor - 1) - \mathbf{P}_q^{(n-1)}(\lceil n/2 \rceil, n-1))). \quad (3)\end{aligned}$$

In the remainder of the paper, in some cases, we will be using the notation $\Delta U(parameter)$ to denote the evaluation of ΔU under a certain value of $parameter$.

The following observation, whose proof is left to the appendix for brevity, will be useful.

Lemma 1. *For any $i \in W$, $\Delta U(p_{\mathcal{C}})$ is a non-decreasing function in $p_{\mathcal{C}} \in [0, 1]$.*

4 Algorithmic Mechanism

Appropriate strategies to carry out the computation with the desired probability of error under various scenarios are considered in this section. It is important to stress again that, in order to obtain a mechanism that is useful for any of those scenarios we do not restrict ourselves to a particular instance of payoffs or reward models leaving those variables as parameters. Thus, we focus our study here on how to choose $p_{\mathcal{A}}$ to have the probability of error bounded by ε for each of the reward models assuming that the payoffs have already been chosen by the master or are fixed exogenously. For settings where payoffs and reward models are a choice of the master, its utility can be easily maximized choosing those parameters conveniently in Equation 2, as demonstrated in Section 5.

In order to design an efficient mechanism, the following issues must be taken into account. Although known, the worker-type distribution is assumed to be arbitrary. Likewise, the particular value of ε is arbitrary given that it is an input of the problem. Finally,

although the priority is to obtain $P_{wrong} \leq \varepsilon$, it is desirable to maximize the utility of the master under such restriction. Thus, the mechanism to choose $p_{\mathcal{A}}$ is designed taking into account two scenarios that we name: *guided rationals* – when a specific behavior of rational workers ($p_{\mathcal{C}}$) has to be enforced – and *free rationals* – otherwise. We analyze these scenarios in the following sections. An explicit protocol implementing this mechanism, is detailed in Algorithm 2.

Algorithm 2: Master protocol to choose $p_{\mathcal{A}}$. $\mathcal{R}_i$ is a Boolean variable indicating if model $\mathcal{R}_i$ is used.

Free rationals:
if $\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2 \rceil, n) > \varepsilon$ **then** /* even if $p_{\mathcal{C}} = 0$, P_{wrong} is big */
    $p_{\mathcal{A}} \leftarrow 1 - \varepsilon / \mathbf{P}^{(n)}_{p_\mu + p_\rho}(\lceil n/2 \rceil, n)$; $q \leftarrow p_\mu + p_\rho$
else if $\mathbf{P}^{(n)}_{p_\mu + p_\rho}(\lceil n/2 \rceil, n) \leq \varepsilon$ **then** /* even if $p_{\mathcal{C}} = 1$, P_{wrong} is low */
    $p_{\mathcal{A}} \leftarrow 0$; $q \leftarrow p_\mu + p_\rho$
else if $\Delta U(p_{\mathcal{C}} = 1, p_{\mathcal{A}} = 0) \leq 0$ **and** $(\mathcal{R}_{\mathrm{m}} \vee \mathcal{R}_{\pm})$ **then** /* $p_{\mathcal{C}} = 0$, even if $p_{\mathcal{A}} = 0$ */
    $p_{\mathcal{A}} \leftarrow 0$; $q \leftarrow p_\mu$

Guided rationals:
    else /* $p_{\mathcal{C}} = 0$ enforced */
        $q \leftarrow p_\mu$
        case $\mathcal{R}_{\mathrm{m}}$
            $p_{\mathcal{A}} \leftarrow 1 - \frac{WP_{\mathcal{C}} + WB_{\mathcal{Y}} - WC_{\mathcal{T}}}{WP_{\mathcal{C}} + WB_{\mathcal{Y}} (\mathbf{P}^{(n-1)}_{p_\mu + p_\rho}(\lfloor n/2 \rfloor, n-1) + \mathbf{P}^{(n-1)}_{p_\mu + p_\rho}(\lceil n/2 \rceil, n-1))}$
        case $\mathcal{R}_{\mathrm{a}} \vee \mathcal{R}_{\emptyset}$
            $p_{\mathcal{A}} \leftarrow \frac{WC_{\mathcal{T}}}{WP_{\mathcal{C}} + WB_{\mathcal{Y}}} + \psi$ /* $\psi > 0$ is an arbitrarily small constant. */
        case $\mathcal{R}_{\pm}$
            $p_{\mathcal{A}} \leftarrow 1 - \frac{WP_{\mathcal{C}} + WB_{\mathcal{Y}} - WC_{\mathcal{T}}}{(WP_{\mathcal{C}} + WB_{\mathcal{Y}})(\mathbf{P}^{(n-1)}_{p_\mu + p_\rho}(\lfloor n/2 \rfloor, n-1) + \mathbf{P}^{(n-1)}_{p_\mu + p_\rho}(\lceil n/2 \rceil, n-1))}$

if $U_M(p_{\mathcal{A}}, q) < U_M(1 - \varepsilon, p_\mu + p_\rho)$ **then** $p_{\mathcal{A}} \leftarrow 1 - \varepsilon$

4.1 Free Rationals

We study in this section the various cases where the behavior of rational workers does not need to be enforced. As mentioned before the main goal is to carry out the computation obtaining the correct output with probability at least $1 - \varepsilon$. Provided that this goal is achieved, it is desirable to maximize the utility of the master. Hence if, for a given instance of the problem, the expected utility of the master utilizing the mechanism described below is smaller than the utility of setting $p_{\mathcal{A}} = 1 - \varepsilon$, the latter is used, because this value trivially guarantees the desired probability of error while yielding better utility.

Lemma 2. *In order to guarantee $P_{wrong} \leq \varepsilon$, it is enough to set $p_{\mathcal{A}} = 1 - \varepsilon$.*

First, we consider pesimistic worker-type distributions, i.e., distributions where p_μ is so large that the probability of having a majority of bad answers is above the desired upper bound, more precisely, when $\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2 \rceil, n) > \varepsilon$. Thus, even if all rationals choose to be honest, the probability of error is too large. Hence, in order to lower P_{wrong}, the master has to audit with a probability big enough, perhaps bigger than the minimum needed to ensure that all rationals are honest. Rational workers still might use some $p_{\mathcal{C}} < 1$ corresponding to some NE. However, as argued later in Theorem 2, the only unique NE that can be obtained in this game is $p_{\mathcal{C}} = 0$ and, if the parameters of the game are such that there is some NE such that $p_{\mathcal{C}} > 0$ there is also another NE in $p_{\mathcal{C}} = 1$. Therefore, to give error-probability guarantees it has to be assumed that all rational workers cheat. Thus, in this case $p_{\mathcal{A}}$ is set from Equation 1 to $1 - \varepsilon / \mathbf{P}^{(n)}_{p_\mu + p_\rho}(\lceil n/2 \rceil, n)$.

Lemma 3. *In order to guarantee $P_{wrong} \leq \varepsilon$, it is enough to set $p_{\mathcal{A}} = 1 - \varepsilon / \mathbf{P}^{(n)}_{p_\mu + p_\rho}(\lceil n/2 \rceil, n)$.*

Now, we consider cases where no audit is needed to achieve the desired bound on the probability of error. The first case occurs when the type-distribution is such that, even if all rational workers cheat, the probability of having a majority of bad answers is at most ε. More precisely, if $\mathbf{P}^{(n)}_{p_\mu + p_\rho}(\lceil n/2 \rceil, n) \leq \varepsilon$, then $p_{\mathcal{A}}$ is set to 0. A second case happens when the particular instance of the parameters of the game force a unique NE such that rationals do not cheat even if they know that the result will not be audited. More precisely, if $\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2 \rceil, n) \leq \varepsilon$ and there is a unique NE in $p_{\mathcal{C}} = 0$ if $p_{\mathcal{A}} = 0$, then $p_{\mathcal{A}}$ can be set to 0. To decide under which parameter conditions this case occurs, we observe the following. Replacing in Eq. 3 the value $p_{\mathcal{A}} = 0$ and the payoffs for each reward model (Table 3), it can be shown that $\Delta U(p_{\mathcal{C}}, p_{\mathcal{A}} = 0)$ is increasing in the interval $p_{\mathcal{C}} \in [0, 1]$ for the $\mathcal{R}_{\mathrm{m}}$ and $\mathcal{R}_{\pm}$ models, and a positive constant for the $\mathcal{R}_{\mathrm{a}}$ and $\mathcal{R}_{\emptyset}$ models. Thus, if $\Delta U(p_{\mathcal{C}} = 1, p_{\mathcal{A}} = 0) \leq 0$ and one of the $\mathcal{R}_{\mathrm{m}}$ and $\mathcal{R}_{\pm}$ models are used, the rate of growth of ΔU implies a single pure NE at $p_{\mathcal{C}} = 0$. In this case, no rational worker cheats and if $\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2 \rceil, n) \leq \varepsilon$ then $p_{\mathcal{A}}$ is set to 0.

Lemma 4. *In order to guarantee $P_{wrong} \leq \varepsilon$, if $\mathbf{P}^{(n)}_{p_\mu + p_\rho}(\lceil n/2 \rceil, n) \leq \varepsilon$, or if $\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2 \rceil, n) \leq \varepsilon$ and there is a unique NE in $p_{\mathcal{C}} = 0$ when $p_{\mathcal{A}} = 0$, it is enough to set $p_{\mathcal{A}} = 0$.*

4.2 Guided Rationals

We study in this section worker-type distributions such that the master can take advantage of a specific NE to achieve the desired bound on the probability of error. Given that the scenario where all players cheat was considered in Section 4.1, in this section it is enough to study Equation 3 for each reward model conditioning $\Delta U(p_{\mathcal{C}} = 1) \leq 0$ to obtain appropriate values for $p_{\mathcal{A}}$. As proved in the following lemma, the specific value $p_{\mathcal{A}}$ assigned depends on the reward model and it is set so that, simultaneously, a unique pure NE is forced at $p_{\mathcal{C}} = 0$ (rendering the rationals truthful) and the error bound is achieved.

Lemma 5. *In order to guarantee $P_{wrong} \leq \varepsilon$, if $\mathbf{P}^{(n)}_{p_\mu+p_\rho}(\lceil n/2 \rceil, n) > \varepsilon$ and $\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2 \rceil, n) \leq \varepsilon$, it is enough to set $p_{\mathcal{A}}$ as follows.*

$$p_{\mathcal{A}} \begin{cases} = 1 - \frac{WP_{\mathcal{C}} + WB_{\mathcal{Y}} - WC_{\mathcal{T}}}{WP_{\mathcal{C}} + WB_{\mathcal{Y}}(\mathbf{P}^{(n-1)}_{p_\mu+p_\rho}(\lfloor n/2 \rfloor, n-1) + \mathbf{P}^{(n-1)}_{p_\mu+p_\rho}(\lceil n/2 \rceil, n-1))} & \text{for } \mathcal{R}_{\mathrm{m}}, \\ > \frac{WC_{\mathcal{T}}}{WP_{\mathcal{C}} + WB_{\mathcal{Y}}} & \text{for } \mathcal{R}_{\mathrm{a}} \text{ and } \mathcal{R}_{\emptyset}, \\ = 1 - \frac{WP_{\mathcal{C}} + WB_{\mathcal{Y}} - WC_{\mathcal{T}}}{(WP_{\mathcal{C}} + WB_{\mathcal{Y}})(\mathbf{P}^{(n-1)}_{p_\mu+p_\rho}(\lfloor n/2 \rfloor, n-1) + \mathbf{P}^{(n-1)}_{p_\mu+p_\rho}(\lceil n/2 \rceil, n-1))} & \text{for } \mathcal{R}_{\pm}. \end{cases}$$

Proof. It was shown in Lemma 1 that, for any of the reward models, $\Delta U(p_{\mathcal{C}})$ is an increasing function in the interval $p_{\mathcal{C}} \in [0, 1]$. Then, in order to enforce a unique NE, it is enough to condition $\Delta U(p_{\mathcal{C}} = 1) \leq 0$ while minimizing the cost of verification. Thus, replacing the payoffs from Table 3 making Equation 3 $\Delta U(p_{\mathcal{C}} = 1) \leq 0$ for each model the claimed values of $p_{\mathcal{A}}$ are obtained.

4.3 Correctness

The following theorem summarizes the previous analyses and proves the correctness of the mechanism designed. Its proof is the simple aggegation of the results presented.

Theorem 1. *The game obtained by combining the parameters of the system with the values of $p_{\mathcal{A}}$ obtained in Sections 4.1 and 4.2 satisfy that $P_{wrong} \leq \varepsilon$.*

4.4 Optimality

In this section we show that only two approaches are feasible to bound the probability of accepting an incorrect value. In this respect, the strategy enforced by the mechanism designed is optimal.

Theorem 2. *In order to achieve $P_{wrong} \leq \varepsilon$, the only feasible approaches are either to enforce a NE where $p_{\mathcal{C}} = 0$ or to use a $p_{\mathcal{A}}$ such that $(1 - p_{\mathcal{A}})\mathbf{P}^{(n)}_{p_\mu+p_\rho}(\lceil n/2 \rceil, n) \leq \varepsilon$.*

Proof. It can be shown as in Lemma 5 that ΔU is increasing for all q, so $\Delta U(p_{\mathcal{C}} < 1) \leq \Delta U(p_{\mathcal{C}} = 1)$. Then, the only NE that can be made unique corresponds to $p_{\mathcal{C}} = 0$ (recall the NE conditions). Consider any other NE where $p_{\mathcal{C}} > 0$ which is not unique. Then $p_{\mathcal{C}} = 1$ is one of these NE. In face of more than one equilibrium to choose from, different players might choose different $p_{\mathcal{C}}$'s. Thus, for the purpose of a worst case analysis with respect to the probability of error, it has to be assumed that all players cheat. But then $p_{\mathcal{A}}$ has to be chosen so that $(1 - p_{\mathcal{A}})\mathbf{P}^{(n)}_{p_\mu+p_\rho}(\lceil n/2 \rceil, n) \leq \varepsilon$.

4.5 Computational Issues

In previous sections, a mechanism for the master to choose appropriate values of $p_{\mathcal{A}}$ for different scenarios was designed. A natural question is what is the computational cost of using such mechanism. In addition to simple arithmetical calculations, there are two kinds of relevant computations required: binomial probabilities and verification of

conditions for Nash equilibria. Both computations are n-th degree polynomial evaluations and can be carried out using any of the well-known numerical tools [21] with polynomial asymptotic cost. These numerical methods yield only approximations, but all these calculations are performed either to decide in which case the parameters fit in, or to assign a value to $p_{\mathcal{A}}$, or to compare utilities. Given that these evaluations and assignments were obtained in the design as inequalities or restricted only to lower bounds, it is enough to choose the appropriate side of the approximation in each case.

Regarding the computational resources that rational workers require to carry out these calculations, notice that the choice of $p_{\mathcal{A}}$ in the mechanism either yields a unique NE in $p_{\mathcal{C}} = 0$ or does not take advantage of the behavior of rational workers. Furthermore, $p_{\mathcal{C}} = 1$ was assumed as a worst case. Notice from Table 3 and Equation 3 that setting $WP_{\mathcal{C}} = WB_{\mathcal{Y}} = 0$ for the later cases we have a dominant strategy in $p_{\mathcal{C}} = 1$. (Recall that $WB_{\mathcal{Y}}$ and $WP_{\mathcal{C}}$ can be chosen by the master.) Thus, the mechanism is enriched so that rational workers are enforced to use always a unique NE, either $p_{\mathcal{C}} = 0$ or $p_{\mathcal{C}} = 1$. Then, in order to make the computation feasible to the workers, the master sends together with the task a "certificate" proving such equilibrium. Such a certificate is the value of $p_{\mathcal{A}}$ and the payoff values, which is enough to verify uniqueness (recall the analysis in Section 4).

5 Putting the Mechanism into Action

In this section two realistic scenarios in which the master-worker model considered could be naturally applicable are proposed. For these scenarios, we determine how to choose $p_{\mathcal{A}}$ and n in the case where the behavior of rational workers is enforced, i.e., under the conditions of Lemma 5.

5.1 SETI-like Scenario

The first scenario considered is a volunteering computing system such as SETI@home, where users accept to donate part of their processors idle time to collaborate in the computation of large tasks. In this case, we assume that workers incur in no cost to perform the task, but they obtain a benefit by being recognized as having performed it (possibly in the form of prestige, e.g., by being included on SETI's top contributors list). Hence, we assume that $WB_{\mathcal{Y}} > WC_{\mathcal{T}} = 0$. The master incurs in a (possibly small) cost $MC_{\mathcal{Y}}$ when rewarding a worker (e.g., by advertising its participation in the project). As assumed in the general model, in this model the master may audit the values returned by the workers, at a cost $MC_{\mathcal{A}} > 0$. We also assume that the master obtains a benefit $MB_{\mathcal{R}} > MC_{\mathcal{Y}}$ if it accepts the correct result of the task, and suffers a cost $MP_{\mathcal{W}} > MC_{\mathcal{A}}$ if it accepts an incorrect value.

Plugging $WC_{\mathcal{T}} = 0$ in the lower bounds of Lemma 5 it can be seen that, for this scenario and conditions, in order to achieve the desired bound on P_{wrong}, it is enough to set $p_{\mathcal{A}}$ to 0 for the $\mathcal{R}_{\mathrm{m}}$ and $\mathcal{R}_{\pm}$ models and arbitrarily close to 0 for the $\mathcal{R}_{\mathrm{a}}$ and $\mathcal{R}_{\emptyset}$ models. So, we want to choose $\delta \leq p_{\mathcal{A}} \leq 1, \delta \to 0$, so that the utility of the master is maximized. However, using calculus, it can be seen that U_M is monotonic in such range. Nevertheless, the growth of such function depends on the specific instance of the

master-payoff parameters. Thus, it is enough to choose one of the extreme values of $p_{\mathcal{A}}$. I.e.,

$$U_M \approx \max\{MB_{\mathcal{R}} - MC_{\mathcal{A}} - n(1-p_\mu)MC_{\mathcal{Y}}, \\ MB_{\mathcal{R}}\mathbf{P}^{(n)}_{p_\mu}(0,\lfloor n/2\rfloor) - MP_{\mathcal{W}}\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2\rceil, n) + \gamma\} \quad (4)$$

Where,

$$\gamma = \begin{cases} -MC_{\mathcal{Y}}(\mathbf{E}^{(n)}_{1-p_\mu}(\lceil n/2\rceil, n) + \mathbf{E}^{(n)}_{p_\mu}(\lceil n/2\rceil, n)) & \text{for the } \mathcal{R}_{\mathrm{m}} \text{ and } \mathcal{R}_{\pm} \text{ models.} \\ -nMC_{\mathcal{Y}} & \text{for the } \mathcal{R}_{\mathrm{a}} \text{ model.} \\ 0 & \text{for the } \mathcal{R}_{\emptyset} \text{ model.} \end{cases}$$

The approximation given in Equation 4 provides a mechanism to choose $p_{\mathcal{A}}$ and n so that U_M is maximized for $P_{wrong} \leq \varepsilon$ for any given worker-type distribution, reward model, and set of payoff parameters in the SETI scenario. For example, given that $\mathbf{E}^{(n)}_{1-p_\mu}(\lceil n/2\rceil, n) + \mathbf{E}^{(n)}_{p_\mu}(\lceil n/2\rceil, n) \geq n(1-p_\mu)$, if $\mathcal{R}_{\mathrm{m}}$ or $\mathcal{R}_{\pm}$ is used and $MP_{\mathcal{W}}\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2\rceil, n) > MC_{\mathcal{A}}$ for some n, the best choice is $p_{\mathcal{A}} = 1$. On the other hand, if $\mathcal{R}_{\emptyset}$ is used, $p_\mu < 1/2$, and $MB_{\mathcal{R}} + MP_{\mathcal{W}} \leq 2MC_{\mathcal{A}} + nMC_{\mathcal{Y}}$, $p_{\mathcal{A}} = 0$ is the best choice. Hence, if the master were to choose the reward model, it is clear that in the above case it would choose $\mathcal{R}_{\emptyset}$. Similar examples can be given for each combination.

5.2 Contractor Scenario

The second scenario considered is a company that buys computational power from Internet users and sells it to computation-hungry costumers. In this case the company pays the users an amount $S = WB_{\mathcal{Y}} = MC_{\mathcal{Y}}$ for using their computing capabilities, and charges the consumers another amount $MB_{\mathcal{R}} > MC_{\mathcal{Y}}$ for the provided service. Since the users are not volunteers in this scenario, we assume that computing a task is not free for them (i.e., $WC_{\mathcal{T}} > 0$), and that rational workers must have incentives to participate (i.e., $U > 0$). As in the previous case, we assume that the master verifies and has a cost for accepting a wrong value, such that $MP_{\mathcal{W}} > MC_{\mathcal{A}} > 0$.

As mentioned before, using calculus it can be seen that U_M is monotonic on $p_{\mathcal{A}}$ but the growth depends on the specific instance of master-payoff parameters. Thus, the maximum expected utility can be obtained for one of the extreme values. Naturally, $p_{\mathcal{A}} = 1$ is the upper bound. For the lower bound, $p_{\mathcal{A}}$ must be appropriately bounded so that the utility of rational workers is positive and $P_{wrong} \leq \varepsilon$. For example, for the $\mathcal{R}_{\emptyset}$ model, using Lemma 5 and conditioning $U > 0$, we get,

$$U_M \approx \max\Bigg\{MB_{\mathcal{R}} - MC_{\mathcal{A}} - n(1-p_\mu)S, \\ \frac{WC_{\mathcal{T}}}{S}\left(MB_{\mathcal{R}} - MC_{\mathcal{A}} - n(1-p_\mu)S\right) \\ + \left(1 - \frac{WC_{\mathcal{T}}}{S}\right)\left(MB_{\mathcal{R}}\mathbf{P}^{(n)}_{p_\mu}(0,\lfloor n/2\rfloor) - MP_{\mathcal{W}}\mathbf{P}^{(n)}_{p_\mu}(\lceil n/2\rceil, n)\right)\Bigg\} \quad (5)$$

As in the previous section, the approximation given in Equation 5, and similar equations for the other reward models which are omitted for clarity, provide a mechanism

to choose $p_{\mathcal{A}}$ and n so that U_M is maximized for $P_{wrong} \leq \varepsilon$ for any given worker-type distribution, reward model, and set of payoff parameters in the contractor scenario. Specific examples can be given for each combination of these parameters either if they are fixed exogenously or by the master.

6 Discussion

In this paper we have combined a classical distributed computing approach (voting) with a game-theoretic one (cost-based incentives and payoffs) that lead to an algorithm that enables a master to reliably obtain a task result despite the co-existence of malicious, altruistic and rational workers. To the best of our knowledge, this is the first work to consider such Internet-based master-worker computations under these assumptions.

Several future directions emanate from this work. For example, in this work we have considered a cost-free, weak version of worker collusion (all rational cheaters and malicious workers return the same incorrect task result). It would be interesting to study more involved collusions, as the ones studied in [1] or [5]. Another interesting extension of our work would be to consider the case in which the network is unreliable, and hence the replies of some workers might not reach the master. This should greatly affect the decision policy and the reward scheme of the master. Finally, in this work, we have considered a single-task one-shot protocol, in which the master decides which task result to accept in one round of message exchange with the workers. It would be interesting to consider several task waves over multiple rounds, that is, view the computation as a *Repeated Game* [27]. The master could use the knowledge gained in the previous rounds to increase its utility and decrease its probability of error in future rounds. Issues such as worker *reputation* could be taken into account.

References

1. I. Abraham, D. Dolev, R. Goden, and J.Y. Halpern. Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In *proc. of PODC 2006*, pp. 53–62, 2006.
2. A. S. Aiyer, L. Alvisi, A. Clement, M. Dahlin, J. Martin, and C. Porth. BAR fault tolerance for cooperative services. In *proc. of SOSP 2005*, pp. 45–58, 2005.
3. D. Anderson. BOINC: A system for public-resource computing and storage. In *proc. of GRID 2004*, pp. 4–10, 2004.
4. M. Babaioff, M. Feldman, and N. Nisan. Mixed Strategies in Combinatorial Agency. In *proc. of WINE 2006*, pp. 353–364, 2006.
5. J. R. Douceur. The Sybil attack. In *proc. of IPTPS 2002*, pp. 251–260, 2002.
6. S. Chien and A. Sinclair. Convergence to approximate Nash equilibria in congestion games. In *proc. of SODA 2007*, pp. 169–178, 2007.
7. G. Christodoulou and E. Koutsoupias. Mechanism design for scheduling. *Bulletin of the EATCS*, 97:39–59, 2009.
8. "Enabling Grids for E-sciencE", `http://www.eu-egee.org`.
9. K. Eliaz. Fault tolerant implementation. *Review of Economic Studies*, 69:589–610, 2002.
10. W. Feller. *An Introduction to Probability Theory and Its Applications.* John Wiley & Sons, 3rd edition, 1968.
11. A. Fernández, Ch. Georgiou, L. Lopez, and A. Santos. Reliably executing tasks in the presence of untrusted processors. In *proc. of SRDS 2006*, pp. 39–50, 2006.
12. A. Fernández Anta, Ch. Georgiou, and M. A. Mosteiro. Designing mechanisms for reliable Internet-based computing. In *proc. of NCA 2008*, pp. 315–324, 2008.
13. I.T. Foster and A. Iamnitchi. On death, taxes, and the convergence of P2P and grid computing. In *proc. of IPTPS 2003*, pp. 118–128, 2003.
14. D. Fotakis. Memoryless facility location in one pass. In *proc. of STACS 2006*, pp. 608–620, 2006.
15. M. Gairing. Malicious Bayesian congestion games. In *proc. of WAOA 2008*, pp. 119–132, 2008.
16. P. Golle and I. Mironov. Uncheatable distributed computations. In *proc. of CT-RSA 2001*, pp. 425–440, 2001.
17. M. Halldorsson, J.Y. Halpern, L. Li, and V. Mirrokni. On spectrum sharing games. In *proc. of PODC 2004*, pp. 107—114, 2004.
18. J.Y. Halpern. Computer science and game theory: A brief survey. *Palgrave Dictionary of Economics*, 2007.
19. J.Y. Halpern and V. Teague. Rational secret sharing and multiparty computation. In *proc. of STOC 2004*, pp. 623–632, 2004.
20. J. C. Harsanyi. Games with incomplete information played by Bayesian players, I, II, III. *Management Science*, 14:159182, 320332, 468502, 1967.
21. W. G. Horner. A new method of solving numerical equations of all orders by continuous approximation. *Philos. Trans. Roy. Soc. London* 109:308–335, 1819.
22. K.M. Konwar, S. Rajasekaran, and A.A. Shvartsman. Robust network supercomputing with malicious processes. In *proc. of DISC 2006*, pp. 474–488, 2006.
23. E. Korpela, D. Werthimer, D. Anderson, J. Cobb, and M. Lebofsky. SETI@home: Massively distributed computing for SETI. *Computing in Science and Engineering*, 3(1):78–83, 2001.
24. E. Koutsoupias and Ch. Papadimitriou. Worst-case equilibria. In *proc. of STACS 1999*, pp. 404–413, 1999.
25. H. C. Li, A. Clement, M. Marchetti, M. Kapritsos, L. Robison, L. Alvisi, and M. Dahlin. FlightPath: Obedience vs. choice in cooperative services. In *proc. of OSDI 2008*, pp. 355–368, 2008.
26. H. C. Li, A. Clement, E. L. Wong, J. Napper, I. Roy, L. Alvisi, and M. Dahlin. BAR gossip. In *proc. of OSDI 2006*, pp. 191–204, 2006.
27. G. Mailath and L. Samuelson. *Repeated Games and Reputations: Long-run Relationships*, Oxford University Press, 2006.

28. M. Mavronicolas and P. Spirakis. The price of selfish routing. *Algorithmica*, 48(1):91–126, 2007.
29. M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.
30. T. Moscibroda, S. Schmid, and R. Wattenhofer. When selfish meets evil: byzantine players in a virus inoculation game. In *proc. of PODC 2006*, pp. 35–44, 2006.
31. J.F. Nash. Equilibrium points in n-person games. *National Academy of Sciences*, 36(1):48–49, 1950.
32. N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.
33. N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007.
34. M. J. Osborne. *An Introduction to Game Theory*. Oxford University Press, 2003.
35. T. Roughgarden and E. Tardos. How bad is selfish routing? *Journal of ACM*, 49(2):236–259, 2002.
36. R. W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2:65-67, 1973.
37. J. Shneidman and D.C. Parkes. Rationality and self-interest in P2P networks. In *proc. of IPTPS 2003*, pp. 139–148, 2003.
38. *Tables of Probability Functions*, Volume = 2, National Bureau of Standards, 1942.
39. 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, 2005.

Appendix

Proof of Lemma 1

Proof. In order to prove this lemma, it is enough to replace the payoffs from Table 3 in Equation 3 for each model as follows.
$\mathcal{R}_{\mathrm{m}}$ model.

$$\begin{aligned}\Delta U = {} & WC_{\mathcal{T}} - WP_{\mathcal{C}} - WB_{\mathcal{Y}} + (1 - p_{\mathcal{A}}) \\ & \big((WP_{\mathcal{C}} + WB_{\mathcal{Y}})\mathbf{P}_q^{(n-1)}(\lfloor n/2 \rfloor, n-1) \\ & + WP_{\mathcal{C}}\mathbf{P}_q^{(n-1)}(0, \lfloor n/2 \rfloor - 1) + WB_{\mathcal{Y}}\mathbf{P}_q^{(n-1)}(\lceil n/2 \rceil, n-1)\big).\end{aligned}$$

$$\begin{aligned}\Delta U = {} & WC_{\mathcal{T}} - WP_{\mathcal{C}} - WB_{\mathcal{Y}} + (1 - p_{\mathcal{A}}) \\ & \big(WP_{\mathcal{C}} + WB_{\mathcal{Y}}(\mathbf{P}_q^{(n-1)}(\lfloor n/2 \rfloor, n-1) + \mathbf{P}_q^{(n-1)}(\lceil n/2 \rceil, n-1))\big).\end{aligned}$$

It can be seen that ΔU is an increasing function in the interval $q \in [0, 1]$ hence the claim follows for this model.
$\mathcal{R}_{\mathrm{a}}$ and $\mathcal{R}_{\emptyset}$ models.

For the $\mathcal{R}_{\mathrm{a}}$ model,

$$\begin{aligned}\Delta U = {} & WC_{\mathcal{T}} - WP_{\mathcal{C}} - WB_{\mathcal{Y}} + (1 - p_{\mathcal{A}}) \\ & (WP_{\mathcal{C}} + WB_{\mathcal{Y}})(\mathbf{P}_q^{(n-1)}(\lfloor n/2 \rfloor, n-1) + \mathbf{P}_q^{(n-1)}(0, \lfloor n/2 \rfloor - 1)).\end{aligned}$$

$$\Delta U = WC_{\mathcal{T}} - WP_{\mathcal{C}} - WB_{\mathcal{Y}} + (1 - p_{\mathcal{A}})(WP_{\mathcal{C}} + WB_{\mathcal{Y}}).$$

And for the $\mathcal{R}_{\emptyset}$ model,

$$\begin{aligned}\Delta U = {} & WC_{\mathcal{T}} - WP_{\mathcal{C}} - WB_{\mathcal{Y}} + (1 - p_{\mathcal{A}}) \\ & \big(WP_{\mathcal{C}}\mathbf{P}_q^{(n-1)}(\lfloor n/2 \rfloor, n-1) + WB_{\mathcal{Y}}\mathbf{P}_q^{(n-1)}(0, \lfloor n/2 \rfloor) \\ & + WP_{\mathcal{C}}\mathbf{P}_q^{(n-1)}(0, \lfloor n/2 \rfloor - 1) + WB_{\mathcal{Y}}\mathbf{P}_q^{(n-1)}(\lceil n/2 \rceil, n-1)\big).\end{aligned}$$

$$\Delta U = WC_{\mathcal{T}} - WP_{\mathcal{C}} - WB_{\mathcal{Y}} + (1 - p_{\mathcal{A}})(WP_{\mathcal{C}} + WB_{\mathcal{Y}}).$$

Thus, ΔU is a constant with respect to $p_{\mathcal{C}}$ hence it is non-decreasing for this model.
$\mathcal{R}_{\pm}$ model.

$$\begin{aligned}\Delta U = {} & WC_{\mathcal{T}} - WP_{\mathcal{C}} - WB_{\mathcal{Y}} + (1 - p_{\mathcal{A}})(WP_{\mathcal{C}} + WB_{\mathcal{Y}}) \\ & \big(\mathbf{P}_q^{(n-1)}(\lfloor n/2 \rfloor, n-1) + \mathbf{P}_q^{(n-1)}(\lceil n/2 \rceil, n-1)\big).\end{aligned}$$

Again, it can be seen that ΔU is an increasing function in the interval $q \in [0, 1]$ hence the claim follows for this model.

$W = \{1, 2, \ldots, n\}$	set of n workers
M	master processor
p_ρ	probability of a worker to be of rational type
p_μ	probability of a worker to be of malicious type
p_a	probability of a worker to be of altruistic type
$p_\mathcal{A}$	probability that the master audits (computes task and checks worker answers)
P_{wrong}	probability that the master obtains a wrong value
ε	desired bound on the probability of error (master not accepting correct answer)
$\{\mathcal{C}, \overline{\mathcal{C}}\}$	action space of a worker
$p_\mathcal{C}$	probability of a worker to cheat
s	strategy profile (a mapping from players to pure strategies)
s_i	strategy used by player i in the strategy profile s
s_{-i}	strategy used by each player but i in the strategy profile s
σ	mixed strategy profile (mapping from players to prob. distrib. over pure strat.)
σ_i	probability distribution over pure strategies used by player i in σ
σ_{-i}	probability distribution over pure strategies used by each player but i in σ
$U_i(s_i, \sigma_{-i})$	expected utility of player i with mixed strategy profile σ
$supp(\sigma_i)$	set of strategies of player i with probability > 0 in σ
ΔU_i or ΔU or $\Delta U(\cdot)$	$U_i(s_i = \mathcal{C}, \sigma_{-i}) - U_i(s_i = \overline{\mathcal{C}}, \sigma_{-i})$
$\mathbf{P}_q^{(n)}(a, b)$	$\sum_{i=a}^{b} \binom{n}{i} q^i (1-q)^{n-i}$

Table 4. Summary of Symbols