Reputation-Based Mechanisms for Evolutionary Master-Worker Computing

Evgenia Christoforou[1,2], Antonio Fernández Anta[1], Chryssis Georgiou[3], Miguel A. Mosteiro[4], and Angel (Anxo) Sánchez[2,5]

[1] Institute IMDEA Networks, Madrid, Spain
[2] Universidad Carlos III de Madrid, Madrid, Spain
[3] University of Cyprus, Nicosia, Cyprus
[4] Kean University, Union, NJ, USA & Univ. Rey Juan Carlos, Madrid, Spain
[5] BIFI Institute, Zaragoza, Spain

Abstract. We consider Internet-based Master-Worker task computing systems, such as SETI@home, where a master sends tasks to potentially unreliable workers, and the workers execute and report back the result. We model such computations using evolutionary dynamics and consider three type of workers: *altruistic*, *malicious* and *rational*. Altruistic workers always compute and return the correct result, malicious workers always return an incorrect result, and rational (selfish) workers decide to be truthful or to cheat, based on the strategy that increases their benefit. The goal of the master is to reach eventual correctness, that is, reach a state of the computation that always receives the correct results. To this respect, we propose a mechanism that uses *reinforcement learning* to induce a correct behavior to rational workers; to cope with malice we employ *reputation schemes*. We analyze our reputation-based mechanism modeling it as a Markov chain and we give provable guarantees under which truthful behavior can be ensured. Simulation results, obtained using parameter values that are likely to occur in practice, reveal interesting trade-offs between various metrics, parameters and reputation types, affecting cost, time of convergence to a truthful behavior and tolerance to cheaters.

Keywords: Volunteer computing, evolutionary game theory, reinforcement learning, reputation.

1 Introduction

Motivation and Prior Work. The need for high-performance computing and the growing use of personal computers and their capabilities (i.e. CPU and GPU), and the wide access to the Internet, have led to the development of Internet-based computing. At present, Internet-based 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 [22] running on the BOINC [4] platform. A profit-seeking computation platform has also been developed by Amazon, called Mechanical Turk [3]. Although the potential is great, the use of Internet-based computing is limited by the untrustworthy nature of the platform's components [4, 18].

R. Baldoni, N. Nisse, and M. van Steen (Eds.): OPODIS 2013, LNCS 8304, pp. 98–113, 2013.

In Internet-based Master-Worker task computing systems a master process sends tasks, across the Internet, to worker processes, that execute and report back the result. However, these workers are not trustworthy, and hence might report incorrect results [4, 5, 20]. Prior work has considered different approaches in tackling the problem. A classical Distributing Computing approach 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 [8] that always return the correct result. Under this view, malicious-tolerant protocols have been considered, e.g., [14, 21, 24], where the master decides on the correct result based on majority voting. A Game-theoretic approach is to assume that workers are *rational* [1, 17, 25], that is, a worker 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). Under this view, incentive-based algorithmic mechanisms have been devised, e.g., [15, 30], that employ reward/punish schemes to "enforce" rational workers to act correctly.

In prior work [16], all three types were considered, and both approaches were combined in order to produce an algorithmic mechanism that provides incentives to rational workers to act correctly, while alleviating the malicious workers' actions. All the solutions described are *one-shot (or stateless)* in the sense that the master decides about the outcome of an interaction with the workers involving a specific task, without using any knowledge gained by prior interactions. In [10], we took advantage of the repeated interactions between the master and the workers, assuming the presence of only rational workers. For this purpose, we studied the *dynamics of evolution* [23] of such master-worker computations through *reinforcement learning* [27] 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, while the workers attempt to increase their benefit. Hence, prior work either considered all three types of workers in one-shot computation, or multi-round interactions assuming only rational workers.

In volunteer computing workers join projects to support a scientific goal and/or to gain prestige [5], while in non-volunteer computing workers expect payment. Whatever the reason, Internet-based computing can not be considered a reliable platform [4, 5, 18, 20]. Thus provable guarantees must be given that the designed mechanism provides a reliable platform, especially in commercial platforms where one can not consider altruistic workers. The existence of all three types of workers must be assumed since workers can have a predefined behavior (malicious or altruistic) or not (rational) as we observe from the survey conducted by BOINC [8] and the behavior of its users [7]. A mechanism must be designed that benefits from the repeated interaction with the workers and thus detaches the knowledge of the distribution over the type of workers from the assumptions (in comparison with [16]).

Our contributions

- We design such an algorithmic mechanism that uses reinforcement learning to induce a correct behavior to rational workers while coping with malice using *reputation*. We consider a centralized reputation scheme controlled by the master that may use three different reputation metrics to calculate each worker's reputation. The first is

adopted from [26], the second, which we introduce, allows for a more drastic change of reputation and the third is inspired by BOINC's reputation scheme [6].
- We analyze our reputation-based mechanism modeling it as a Markov chain and we identify conditions under which truthful behavior can be ensured. We analytically prove that by using the second reputation type (introduced in this work for the first time) reliable computation is eventually achieved.
- Simulation results, obtained using parameter values that are likely to occur in practice, reveal interesting trade-offs between various metrics and parameters, such as cost, time of convergence to a truthful behavior, tolerance to cheaters and the type of reputation metric employed. Simulations also reveal better performance of our reputation type (second type) in several realistic cases.

Background and related work. As part of our mechanism we use reinforcement learning to induce the correct behavior of rational workers. Reinforcement learning [27] models how system entities, or *learners*, interact with the environment to decide upon a strategy, and use their experience to select or avoid actions according to the consequences observed. Positive payoffs increase the probability of the strategy just chosen, and negative payoffs reduce this probability. Payoffs are seen as parameterizations of players' responses to their experiences. There are several models of reinforcement learning. A well-known model is that of Bush and Mosteller [9]; this is an aspiration-based reinforcement learning model where negative effects on the probability distribution over strategies are possible, and learning does not fade with time. The learners adapt by comparing their experience with an *aspiration* level. In our work we adapt this reinforcement learning model and we consider a simple aspiration scheme where aspiration is fixed by the workers and does not change during the evolutionary process.

The master reinforces its strategy as a function of the reputation calculated for each worker. Reputation has been widely considered in on-line communities that deal with untrustworthy entities, such as online auctions (e.g., eBay) or P2P file sharing sites (e.g., BitTorrent); it provides a mean of evaluating the degree of trust of an entity [19]. Reputation measures can be characterized in many ways, for example, as objective or subjective, centralized or decentralized. An objective measure comes from an objective assessment process while a subjective measure comes from the subjective belief that each evaluating entity has. In a centralized reputation scheme a central authority evaluates the entities by calculating the ratings received from each participating entity. In a decentralized system entities share their experience with other entities in a distributed manner. In our work, we use the master as a central authority that objectively calculates the reputation of each worker, based on its interaction with it; this centralized approach is also used by BOINC.

The BOINC system itself uses a form of reputation [6] for an optional policy called adaptive replication. This policy avoids replication in the event that a job has been sent to a highly reliable worker. The philosophy of this reputation scheme is to require a long time for the worker to gain a good reputation but a short time to lose it. Our proposed mechanism differs significantly from the one that is used in BOINC. One important difference is that we use auditing to check the validity of the worker's answers while BOINC uses only replication; in this respect, we have a more generic mechanism that also guarantees reliability of the system. Notwithstanding inspired by the way BOINC

handles reputation we have designed a BOINC-like reputation type in our mechanism (called type three).

Sonnek et al. [26] use an adaptive reputation-based technique for task scheduling in volunteer setting (i.e., projects running BOINC). Reputation is used as a mechanism to reduce the degree of redundancy while keeping it possible for the master to verify the results by allocating more reliable nodes. In our work we do not focus on scheduling tasks to more reliable workers to increase reliability but rather we design a mechanism that forces the system to evolve to a reliable state. We also demonstrate several tradeoff between reaching a reliable state fast and master's cost. We have created a reputation function (called reputation type 1) that is analogous to the reputation function used in [26] to evaluate this function's performance in our setting.

Aiyer et al. [2] introduced the BAR model to reason about systems with Byzantine (malicious), Altruistic, and Rational participants. They introduced the notion of *BAR-tolerant* protocols, i.e., protocols that are resilient to both Byzantine faults and rational manipulation. As an application, they designed a cooperative backup service for P2P systems, based on a BAR-tolerant replicated state machine. More recent works have considered other problems in the BAR model (e.g., data transfer [29]). Although the objectives and the model considered are different, our reputation-based mechanism can be considered, in some sense, to be BAR-tolerant.

2 Model

In this section we characterize our model and we present the concepts of auditing, payoffs, rewards and aspiration. We also give a formal definition of the three reputation types used by our mechanism.

Master-Worker Framework. We consider a master and a set W of n workers. The computation is broken into *rounds*, and in each round the master sends a task to the workers to compute and return the result. Based on the workers' replies, the master must decide which is the value most likely to be the correct result for this round. We assume that tasks have a unique solution; although such limitation reduces the scope of application of the presented mechanism [28], there are plenty of computations where the correct solution is unique: e.g., any mathematical function.

Eventual Correctness. The goal of the master is to eventually obtain a reliable computational platform: After some finite number of rounds, the system must guarantee that the master (with minimal cost) obtains the correct task results in every round with probability 1. We call such property *eventual correctness*.

Worker Types. We consider three type of workers: *rational, altruistic* and *malicious*. Rational workers are selfish in a game-theoretic sense and their aim is to maximize their utility (benefit). In the context of this paper, a worker is *honest* in a round, when it truthfully computes and returns the correct result, and it *cheats* when it returns some incorrect value. Altruistic and malicious workers have a predefined behavior, to always be honest or cheat, respectively. Instead, a rational worker decides to be honest or cheat depending on which strategy maximizes its utility. We denote by $p_{Ci}(r)$ the probability of a rational worker i cheating in round r. This probability is not fixed and the worker adjusts it over the course of the computation. The master is not aware of the worker types, neither of a distribution of types (our mechanism does not rely on any statistical information).

While workers make their decision individually and with no coordination, following [24] and [14], we assume that all the workers that cheat in a round return the same incorrect value; this yields a worst case scenario (and hence analysis) for the master with respect to obtaining the correct result using mechanisms where the result is the outcome of voting. It subsumes models where cheaters do not necessarily return the same answer. (This can be seen as a weak form of collusion.)

For simplicity, unless otherwise stated, we assume that workers do not change their type over time. Observe that in practice it is possible that changes occur. For example, a rational worker might become malicious due to a bug, or a malicious worker (e.g., a worker under the influence of a virus) become altruistic (e.g., if an antivirus software reinstates it). If this may happen, then all our results still apply for long enough periods between two changes.

Auditing, Payoffs, Rewards and Aspiration. To induce the rational workers to be honest, the master employs, when necessary, *auditing* and *reward/punish* schemes. The master, in a round, might decide to audit the response of the workers, at a cost. In this work, auditing means that the master computes the task by itself, and checks which workers have been honest. We denote by $p_{\mathcal{A}}(r)$ the probability of the master auditing the responses of the workers in round r. The master can change this auditing probability over the course of the computation, but restricted to a minimum value $p_{\mathcal{A}}^{min} > 0$. When the master audits, it can accurately reward and punish workers. When the master does not audit, it rewards only those in the weighted majority (see below) of the replies received and punishes no one.

In this work we consider three worker payoff parameters: (a) $WP_{\mathcal{C}}$: worker's punishment for being caught cheating, (b) $WC_{\mathcal{T}}$: worker's cost for computing a task, and (c) $WB_{\mathcal{Y}}$: worker's benefit (typically payment) from the master's reward. Also, following [9], we assume that, in every round, a worker i has an *aspiration* a_i: the minimum benefit it expects to obtain in a round. In order to motivate the worker to participate in the computation, the master usually ensures that $WB_{\mathcal{Y}} \geq a_i$; in other words, the worker has the potential of its aspiration to be covered. We assume that the master knows the aspirations. Finally, we assume that the master has the freedom of choosing $WB_{\mathcal{Y}}$ and $WP_{\mathcal{C}}$ with goal of eventual correctness.

Reputation. The reputation of each worker is measured by the master; a centralized reputation mechanism is used. In fact, the workers are unaware that a reputation scheme is in place, and their interaction with the master does not reveal any information about reputation; i.e., the payoffs do not depend on a worker's reputation.

In this work, we consider three reputation metrics. The first one, called *type 1* is analogous to a reputation metric used in [26] and the third one, called *type 3* is inspired by BOINC. We also define our own type called *type 2* that is not influenced by any other reputation type, and as we show in Section 4 it possesses beneficial properties. In all types, the reputation of a worker is determined based on the number of times it was found truthful. Hence, the master may update the reputation of the workers only when it audits. We denote by $aud(r)$ the number of rounds the master audited up to round r, and by $v_i(r)$ we refer to the number of auditing rounds in which worker i was found truthful up to round r. We let $\rho_i(r)$ denote the *reputation* of worker i after round r, and for a given set of workers $Y \subseteq W$ we let $\rho_Y(r) = \sum_{i \in Y} \rho_i(r)$ be the aggregated reputation of the workers in Y, by aggregating we refer to summing the reputation values. Then, the three reputation types we consider are the following:

Type 1: $\rho_i(r) = (v_i(r) + 1)/(aud(r) + 2)$.
Type 2: $\rho_i(r) = \varepsilon^{aud(r)-v_i(r)}$, for $\varepsilon \in (0, 1)$, when $aud(r) > 0$, and $\rho_i(r) = 1/2$, otherwise.
Type 3: Here we define $\beta_i(r)$ as the error rate of worker i at round r and by $A = 0.05$ the error bound. Reputation for this type is calculated as follows:

Step 1:
$\beta_i(r) \leftarrow 0.1$
if *worker truthful* **then**
  $\beta_i(r) \leftarrow \beta_i(r) \cdot 0.95$ \\ *calculating error rate*
else $\beta_i(r) \leftarrow \beta_i(r) + 0.1$

Step 2:
if $\beta_i(r) > A$ **then**
  $\rho_i(r) \leftarrow 0.001$ \\ *calculating reputation*
else $\rho_i(r) \leftarrow 1 - \sqrt{\frac{\beta_i(r)}{A}}$

In each round, when the master *does not audit*, the result is obtained from the *weighted majority* as follows. Consider a round r. Let $F(r)$ denote the subset of workers that returned an incorrect result, i.e., the rational workers who chose to cheat plus the malicious ones; recall that we assume as a worst case that all cheaters return the same value. Then, $W \setminus F(r)$ is the subset of workers that returned the correct value, i.e., the rational workers who chose to be truthful plus the altruistic ones. Then, if $\rho_{W \setminus F(r)}(r) > \rho_{F(r)}(r)$, the master will accept the correct value, otherwise it will accept an incorrect value. The mechanism, presented in the next section, employs auditing and appropriate incentives so that rational workers become truthful with high reputation, while malicious workers (alternatively altruistic workers) end up having very low (altr. very high) reputation after a few auditing rounds.

3 Reputation-Based Mechanism

We now present our reputation-based mechanism. The mechanism is composed by an algorithm run by the master and an algorithm run by each worker.

Master's Algorithm. The algorithm begins by choosing the initial probability of auditing and the initial reputation (same for all workers). The initial probability of auditing will be set according to the information the master has about the environment (e.g., workers' initial p_C). For example, if it has no information about the environment, a possibly safe approach is to initially set $p_{\mathcal{A}} = 0.5$. The master also chooses the reputation type to use (e.g., type 1, 2 or 3).

After that, at each round, the master sends a task to all workers and, when all answers are received, the master audits the answers with probability $p_{\mathcal{A}}$. In the case the answers are not audited, the master accepts the value returned by the weighed majority, and continues to the next round with the same probability of auditing and the same reputation values for each worker. In the case the answers are audited, the value $p_{\mathcal{A}}$ of the next round is reinforced (i.e., modified according to the accumulated reputation of the cheaters) and the reputations of the workers are updated based on their responses. Then, the master rewards/penalizes the workers appropriately. Specifically, if the master audits and a worker i is a cheater (i.e., $i \in F$), then $\Pi_i = -WP_{\mathcal{C}}$; if i is honest, then $\Pi_i = WB_{\mathcal{Y}}$. If the master does not audit, and i returns the value of the weighted majority (i.e., $i \in W_m$), then $\Pi_i = WB_{\mathcal{Y}}$, otherwise $\Pi_i = 0$.

We include a threshold, denoted by τ, that represents the master's *tolerance* to cheating (typically, we will assume $\tau = 1/2$ in our simulations). If the ratio of the aggregated reputation of cheaters with respect to the total is larger than τ, $p_{\mathcal{A}}$ is increased, and decreased otherwise. The amount by which $p_{\mathcal{A}}$ changes depends on the difference

Algorithm 1. Master's Algorithm

$p_{\mathcal{A}} \leftarrow x$, *where* $x \in [p_{\mathcal{A}}^{min}, 1]$
$aud = 0$
// initially all workers have the same reputation
$\forall i \in W : v_i = 0; \rho_i = 0.5$
for $r \leftarrow 1$ **to** ∞ **do**
  send *a task* T *to all workers in* W
  upon *receiving all answers* **do**
    audit *the answers with probability* $p_{\mathcal{A}}$
    if *the answers were not audited* **then**
      // weighted majority, coin flip in case of a tie
      accept *the value returned by workers in* $W_m \subseteq W$, *where* $\rho_{W_m} > \rho_{W \setminus W_m}$
    else *// the master audits*
      $aud \leftarrow aud + 1$
      Let $F \subseteq W$ *be the set of workers that cheated.*
      $\forall i \in W$:
        if $i \notin F$ **then** $v_i \leftarrow v_i + 1$ *// honest workers*
        update *reputation* ρ_i *of worker* i
      $p_{\mathcal{A}} \leftarrow \min\{1, \max\{p_{\mathcal{A}}^{min}, p_{\mathcal{A}} + \alpha_m(\frac{\rho_F}{\rho_W} - \tau)\}\}$
    $\forall i \in W$: **return** *payoff* Π_i *to worker* i

Algorithm 2. Algorithm for Rational Worker i

$p_{Ci} \leftarrow y$, *where* $y \in [0, 1]$
for $r \leftarrow 1$ **to** ∞ **do**
  receive *a task* T *from the master*
  $S_i \leftarrow -1$ *with probability* p_{Ci},
  and $S_i \leftarrow 1$ *otherwise*
  if $S_i = 1$ **then**
    $\sigma \leftarrow compute(T)$,
  else
    $\sigma \leftarrow$ *arbitrary solution*
  send *response* σ *to the master*
  get payoff Π_i
  if $S_i = 1$ **then**
    $\Pi_i \leftarrow \Pi_i - WC_{\mathcal{T}}$
  $p_{Ci} \leftarrow \max\{0, \min\{1, p_{Ci} - \alpha_w(\Pi_i - a_i)S_i\}\}$

between these values, modulated by a *learning rate* α_m. This latter value determines to what extent the newly acquired information will override the old information. (For example, if $\alpha_m = 0$ the master will never adjust $p_{\mathcal{A}}$.) A pseudocode of the algorithm described is given as Algorithm 1.

Workers' Algorithm. This algorithm is run only by rational workers (recall that altruistic and malicious workers have a predefined behavior).[1] The execution of the algorithm begins with each rational worker i deciding an initial probability of cheating p_{Ci}. In each round, each worker receives a task from the master and, with probability $1 - p_{Ci}$ computes the task and replies to the master with the correct answer. Otherwise, it fabricates an answer, and sends the incorrect response to the master. We use a flag S_i to model the stochastic decision of a worker i to cheat or not. After receiving its payoff, each worker i changes its p_{Ci} according to payoff Π_i, the chosen strategy S_i, and its aspiration a_i.

The workers have a *learning rate* α_w. In this work, we assume that all workers have the same learning rate, that is, they learn in the same manner (see also the discussion in [27]; the learning rate is called step-size there); note that our analysis can be adjusted to accommodate also workers with different learning rates. A pseudocode of the algorithm is given as Algorithm 2.

4 Analysis

We now analyze the reputation-based mechanism. We model the evolution of the mechanism as a Markov Chain, and then discuss the necessary and sufficient conditions for achieving eventual correctness. Modeling a reputation-based mechanism as a Markov Chain is more involved than previous models that do not consider reputation (e.g. [10]).

[1] Since the workers are not aware that a reputation scheme is used, this algorithm is the one considered in [10]; we describe it here for self-containment.

The Markov Chain. Let the state of the Markov chain be given by a vector s. The components of s are: for the master, the probability of auditing $p_{\mathcal{A}}$ and the number of audits before state s, denoted as aud; and for each *rational* worker i, the probability of cheating p_{Ci} and the number of *validations* (i.e., the worker was honest when the master audited) before state s, denoted as v_i. To refer to any component x of vector s we use $x(s)$. Then, $s = \langle p_{\mathcal{A}}(s), aud(s), p_{C1}(s), p_{C2}(s), \ldots, p_{Cn}(s), v_1(s), v_2(s), \ldots, v_n(s)\rangle$.

In order to specify the transition function, we consider the execution of the protocol divided in rounds. In each round, probabilities and *counts* (i.e. numbers of validations and audits) are updated by the mechanism as defined in Algorithms 1 and 2. The state at the end of round r is denoted as s_r. Abusing the notation, we will use $x(r)$ instead of $x(s_r)$ to denote component x of vector s_r. The workers' decisions, the number of cheaters, and the payoffs of each round $r > 0$ are the stochastic outcome of the probabilities and counts at the end of round $r-1$. We specify the transition from s_{r-1} to s_r by the actions taken by the master and the workers during round r.

In the definition of the transition function that follows, the probabilities are limited to $p_{\mathcal{A}}(s) \in [p_{\mathcal{A}}^{min}, 1]$ and for each rational worker i to $p_{Ci}(s) \in [0,1]$, for any state s. The initial state s_0 is arbitrary but restricted to the same limitations. Let $P_F(r)$ be the probability that the set of cheaters in round r is exactly $F \subseteq W$. (That is, $P_F(r) = \prod_{j\in F} p_{Cj}(r-1) \prod_{k\notin F}(1-p_{Ck}(r-1))$.) Then, the transition from state s_{r-1} to s_r is as follows.

- Malicious workers always have $p_C = 1$ and altruistic workers always have $p_C = 0$.
- With probability $p_{\mathcal{A}}(r-1)\cdot P_F(r)$, the master audits when the set of cheaters is F. Then, according to Algorithms 1 and 2, the new state is as follows.

  For the master: $p_{\mathcal{A}}(r) = p_{\mathcal{A}}(r-1) + \alpha_m\,(\rho_F(r)/\rho_W(r) - \tau)$ and $aud(r) = aud(r-1)+1$.

  (1) For each worker $i \in F$: $v_i(r) = v_i(r-1)$ and, if i is rational, then $p_{Ci}(r) = p_{Ci}(r-1) - \alpha_w(a_i + WP_{\mathcal{C}})$.

  (2) For each worker $i \notin F$: $v_i(r) = v_i(r-1)+1$ and, if i is rational, then $p_{Ci}(r) = p_{Ci}(r-1) + \alpha_w(a_i - (WB_{\mathcal{Y}} - WC_{\mathcal{T}}))$.
- With probability $(1-p_{\mathcal{A}}(r-1))P_F(r)$, the master does not audit when the set of cheaters is F. Then, according to Algorithms 1 and 2, the following updates are carried out.

  For the master: $p_{\mathcal{A}}(r) = p_{\mathcal{A}}(r-1)$ and $aud(r) = aud(r-1)$.

  For each worker $i \in W$: $v_i(r) = v_i(r-1)$.

  For each rational worker $i \in F$,

  (3) if $\rho_F(r) > \rho_{W\setminus F}(r)$ then $p_{Ci}(r) = p_{Ci}(r-1) + \alpha_w(WB_{\mathcal{Y}} - a_i)$,

  (4) if $\rho_F(r) < \rho_{W\setminus F}(r)$ then $p_{Ci}(r) = p_{Ci}(r-1) - \alpha_w \cdot a_i$,

  For each rational worker $i \notin F$,

  (5) if $\rho_F(r) > \rho_{W\setminus F}(r)$ then $p_{Ci}(r) = p_{Ci}(r-1) + \alpha_w(a_i + WC_{\mathcal{T}})$,

  (6) if $\rho_F(r) < \rho_{W\setminus F}(r)$ then $p_{Ci}(r) = p_{Ci}(r-1) + \alpha_w(a_i - (WB_{\mathcal{Y}} - WC_{\mathcal{T}}))$.

Recall that, in case of a tie in the weighted majority, the master flips a coin to choose one of the answers, and assigns payoffs accordingly. If that is the case, transitions (3)–(6) apply according to that outcome.

Conditions for Eventual Correctness. We show now the conditions under which the system can guarantee eventual correctness. The analysis is carried out for a universal class of reputation functions characterized by the following properties.

Property 1: For any $X \subset W$ and $Y \subset W$, if the Markov chain evolves in such a way that $\forall i \in X, \lim_{r\to\infty}(v_i(r)/aud(r)) = 1$ and $\forall j \in Y, \lim_{r\to\infty}(v_j(r)/aud(r)) = 0$, then there is some r^* such that $\forall r > r^*, \rho_X(r) > \rho_Y(r)$.

Property 2: For any $X \subset W$ and $Y \subset W$, if $aud(r+1) = aud(r) + 1$ and $\forall j \in X \cup Y$ it is $v_j(r+1) = v_j(r) + 1$ then $\rho_X(r) > \rho_Y(r) \Rightarrow \rho_X(r+1) > \rho_Y(r+1)$.

Observe that all reputation functions (type 1, type 2 and type 3) we consider (cf. Section 2), satisfy Property 1. However, regarding Property 2, while reputation type 2 satisfies it, reputation type 1 and 3 do not. As we show below, this *makes a difference* with respect to guaranteeing eventual correctness.

The following terminology will be used throughout. For any given state s, a set X of workers is called a *reputable set* if $\rho_X(r) > \rho_{W\setminus X}(r)$. In any given state s, let a worker i be called an *honest worker* if $p_{Ci}(s) = 0$. Let a state s be called a *truthful state* if the set of honest workers in state s is reputable. Let a *truthful set* be any set of truthful states. Let a worker be called a *covered worker* if the payoff of returning the correct answer is at least its aspiration plus the computing cost. I.e., for a covered worker i, it is $WB_{\mathcal{Y}} \geq a_i + WC_{\mathcal{T}}$. We refer to the opposite cases as *uncovered worker* ($WB_{\mathcal{Y}} < a_i + WC_{\mathcal{T}}$), *cheater worker* ($p_{Ci}(s) = 1$), *untruthful state* (the set of cheaters in that state is reputable), and *untruthful set*, respectively. Let a set of states S be called *closed* if, once the chain is in any state $s \in S$, it will not move to any state $s' \notin S$. (A singleton closed set is called an *absorbing* state.) For any given set of states S, we say that the chain *reaches* (resp. *leaves*) the set S if the chain reaches some state $s \in S$ (resp. reaches some state $s \notin S$).

In the master's algorithm, a non-zero probability of auditing is always guaranteed. This is a necessary condition. Otherwise, unless the altruistic workers outnumber the rest, a closed untruthful set is reachable, as we show in [11].

Eventual correctness follows if we can show that the Markov chain always ends in a closed truthful set. We prove first that having at least one worker that is altruistic or covered rational is necessary for a closed truthful set to exist. Then we prove that it is also sufficient.

Lemma 1. *If all workers are malicious or uncovered rationals, no truthful set S is closed, if the reputation type satisfies Property 2.*

Proof. Let us consider some state s of a truthful set S. Let Z be the set of honest workers in s. Since s is truthful , then Z is reputable. Since there are no altruistic workers, the workers in Z must be uncovered rational. Let us assume that being in state s the master audits in round r. From Property 2, since all nodes in Z are honest in r, Z is reputable after r. From transition (2), after round r, each worker $i \in Z$ has $p_{Ci}(r) > 0$. Hence, the new state is not truthful , and S is not closed.

Lemma 2. *If at least one worker is altruistic or covered rational, a truthful set S is reachable from any initial state, if the reputation type satisfies Properties 1 and 2.*

Proof (Proof Sketch). Let C be the set of workers that are altruistic or covered rational. From any initial state, there is a non-zero probability that the master audits in all

subsequent rounds. Then, from transition (2) and Properties 1 and 2, there is a non-zero probability of reaching a truthful state s^* in which (a) all workers in C are honest and (b) C is reputable. Once s^* is reached, all subsequent states satisfy these two properties (which define the set S), independently of whether the master audits (from transition (2) and (6), and Property 2).

Now, putting together Lemmas 1 and 2 we obtain the following theorem.

Theorem 1. *Having at least one worker altruistic or covered rational is necessary and sufficient to eventually reach a truthful set S from any initial state, and hence to guarantee eventual correctness, if the reputation type satisfies Properties 1 and 2.*

Observe that if there is no knowledge on the distribution of the workers among the three types (altruistic, malicious and rationals), the only strategy to make sure eventual correctness is achieved, if possible, is to cover all workers. Of course, if all workers are malicious (an unlikely situation, as shown in [8, 12, 13]) there is no possibility of reaching eventual correctness.

5 Simulations

This section complements our analytical results with illustrative simulations. The graphical representation of the data obtained captures the tradeoffs between reliability and cost and among all three reputation types, concepts not visible through the analysis. Here we present simulations for a variety of parameter combinations likely to occur in practice (extracted from [12,13]) and similar to our earlier work [10]. We have designed our own simulation setup by implementing our mechanism (the master's and the workers' algorithms, including the three types of reputation discussed above) using C++. The simulations were contacted on a dual-core AMD Opteron 2.5GHz processor, with 2GB RAM running CentOS version 5.3. We consider a total of 9 workers (that will be rational, altruistic or malicious in the different experiments). The figures represent the average over 10 executions of the implementation, unless otherwise stated (when we show the behavior of typical, individual realizations). The chosen parameters are indicated in the figures. Note that, for simplicity, we consider that all workers have the same aspiration level $a_i = 0.1$, although we have checked that with random values the results are similar to those presented here, provided their variance is not very large. We consider the same learning rate for the master and the workers, i.e., $\alpha = \alpha_m = \alpha_w = 0.1$. Note that the learning rate, as discussed for example in [27] (called step-size there), is generally set to a small constant value for practical reasons. Finally we set $\tau = 0.5$ (see [10]), $p_{\mathcal{A}}^{min} = 0.01$ and $\epsilon = 0.5$ in reputation type 2.

The contents of this section can be summarized as follows: In the next paragraph we present results considering only rational workers and, subsequently, results involving all three type of workers. We continue with a discussion on the number of workers that must be covered and how the choice of reputation affects this. Finally, we briefly show that our mechanism is robust even in the event of having workers changing their behavior (e.g. rational workers becoming malicious due to software or hardware error). An exhaustive account of simulation results is presented in [11].

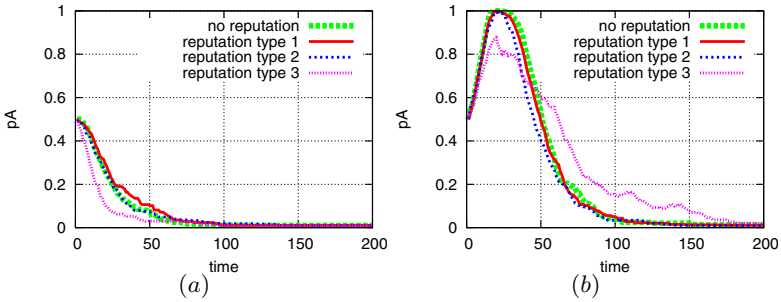

Fig. 1. Rational workers. Auditing probability of the master as a function of time (number of rounds) for parameters $p_{\mathcal{A}} = 0.5$, $\alpha = 0.1$, $a_i = 0.1$, $WB_{\mathcal{Y}} = 1$, $WP_{\mathcal{C}} = 0$ and $WC_{\mathcal{T}} = 0.1$. (a) initial $p_C = 0.5$ (b) initial $p_C = 1$.

Rational Workers. While the main reason for introducing reputation was to cope with malicious workers, as a first step we checked whether reputation improves the algorithm performance for rational workers only. In this case as we see in Figure 1, the first two reputation types give similar results as in the case of no reputation. Reputation type 3, on the other hand, seems to perform better in the case that the initial $p_C = 0.5$, while in the case of $p_C = 1$ the system has a slower convergence rate, but the auditing probability at the first 50 rounds is lower. This has to do with the fact that in type 3 the reputation of a worker is reinforced indirectly, what is directly reinforced depending on the workers honesty is the error rate. Our observations in Figure 1 reveals an interesting tradeoff: depending on whether the master has information on the workers' initial behavior or on the auditing that is willing to perform it will choose to use or not reputation type 3. From Figure 1 we can also see that the mechanism of [10] is enough to bring rational workers to produce the correct output, precisely because of their rationality. Although Figure 1 depicts the $p_{\mathcal{A}}$ of the master and not the p_C of the workers we have observed (see [11]) that for all the initial p_C studied, by the time the master's auditing probability reaches $p_{\mathcal{A}}^{min}$, the system had already reached eventual correctness.

Figure 2 allows to compare the behavior of the three reputation types, with reputation ratio defined as $\sum_{i \in W} \rho_i S_i / |W|$. Reputation type 1 leads rational workers to reputation values close to 1 (at a rate that depends on the value of the initial p_C). However, when type 2 is applied reputation takes values between (0,0.3). This happens because when the master catches a worker cheating, its reputation decreases exponentially, never increasing again. Reputation type 3, on the other hand, allows for dramatic increases and decreases of reputation. This is a result of the indirect way we calculate reputation type 3, as we mentioned above.

Different Types of Workers. We now move to our main case of interest and include different types of workers in our experiments. Figure 3 shows results for the extreme case, with malicious workers, no altruistic workers, and rational workers that initially cheat with probability $p_C = 1$. We observe that if the master does not use reputation and a majority of malicious workers exist, then the master is enforced by the mechanism to audit in every round. Even with a majority of rational workers, it takes a long time for the master to reach $p_{\mathcal{A}}^{min}$, if reputation is not used. Introducing reputation can indeed cope with the challenge of having a majority of malicious workers. For type 1, the larger the

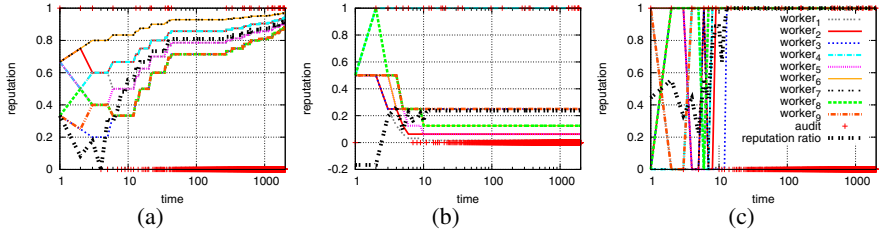

Fig. 2. Rational workers, for an individual realization with initially $p_C = 0.5$, $p_{\mathcal{A}} = 0.5$, $WC_{\mathcal{T}} = 0.1$, $WP_{\mathcal{C}} = 0$, $\alpha = 0.1$ and $a_i = 0.1$. Left, reputation type 1. Middle, reputation type 2. Right, reputation type 3.

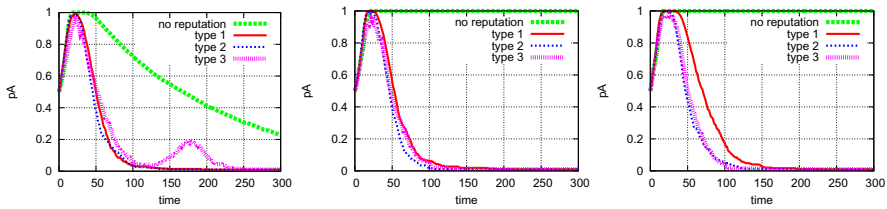

Fig. 3. Master's auditing probability as a function of time in the presence of rational and malicious workers. Parameters in all plots, rationals' initial $p_C = 1$, master's initial $p_{\mathcal{A}} = 0.5$, $WC_{\mathcal{T}} = 0.1$, $WP_{\mathcal{C}} = 0$ and $\alpha = 0.1$, $a_i = 0.1$. In (a) 4 malicious and 5 rationals, (b) 5 malicious and 4 rationals , (c) 8 malicious and 1 rational.

number of malicious workers, the slower the master reaches $p_{\mathcal{A}}^{min}$. On the contrary, the time to convergence to the $p_{\mathcal{A}}^{min}$ is independent of the number of malicious workers for reputation type 2. This is due to the different dynamical behavior of the two reputations discussed above. For reputation type 3, if a majority of rationals exists then convergence is slower. This is counter-intuitive, but as we mentioned before it is linked to the way reputation and error rate are calculated. On the other hand, with type 3, $p_{\mathcal{A}}$ is slightly lower in the first rounds. We thus conclude that reputation type 2 gives better results, as long as at least one rational worker exists and the master is willing to audit slightly more in the first rounds. We have checked that if rational workers are replaced by altruistic ones, the performance of the two reputation schemes improves, as expected.

Covering only a Subset of Rational Workers. In the previous paragraphs we considered only cases where the master was covering all workers, that is, $WB_{\mathcal{Y}} > a + WC_{\mathcal{T}}$ for all workers. For the case with malicious workers, as explained in Section 4, this is unavoidable. But for the case with rational workers, as was argued in the same section, we may avoid covering all workers, a scenario which we now explore. In Figure 4(a) the extreme case of only one covered worker is presented. We see that with reputation type 1 our system does not converge, *which is consistent with the results of* Section 4. The master tolerates a significant percentage of cheaters (since $\tau = 0.5$), creating a very unstable system, where the probability of the master receiving the correct answer varies greatly. This occurs because by tolerating more cheaters, the master creates a

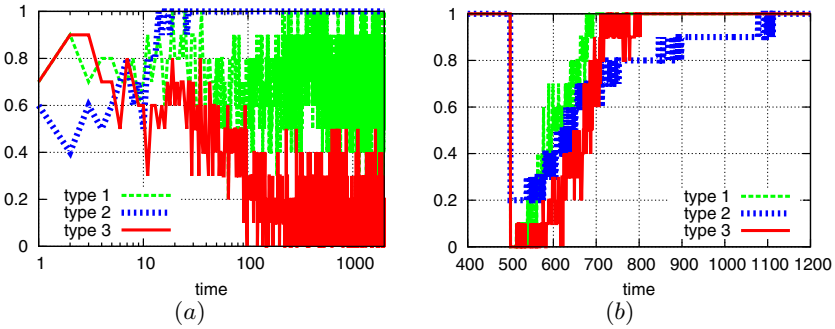

Fig. 4. Correct reply rate as a function of time. Parameters are initial $p_{\mathcal{A}} = 0.5$ $WC_{\mathcal{T}} = 0.1$, $WP_{\mathcal{C}} = 0$ and $\alpha = 0.1$, $a_i = 0.1$. (a) Reputation types 1 and 3 have initial $p_C = 0.5$, while in type 2, $p_C = 1$. (b) initial $p_C = 1$.

system where the cheating probability of the uncovered workers spikes between zero and one. We have found that introducing punishment or reducing the tolerance do not fix this (see [11]). Basically, the reason is that the reputation of honest workers does not always exceed the reputation of cheaters, as indicated by the reputation ratio. In fact, we have checked that for the covered worker, p_C vanishes eventually, but for uncovered workers this is not the case: their p_C takes values usually below 0.6 but close to that value. Given that uncovered workers' p_C is greater than zero, it may occur that the master does not audit and a number of uncovered workers, with reputation higher than the rest, cheat. Because of this, even if the master maintains a high auditing probability, eventual correctness is not guaranteed.

For reputation type 3 our system does not converge, *which is consistent with the results of* Section 4 since reputation type 3 does not satisfy Property 2. Type 3 gives even worse results than type 1, since the correct reply ratio is always lower compared to type 1. Finally, Figure 4(a) shows that our system always converges using reputation type 2, as expected by the analysis in Section 4. A collection of elaborative simulation figures (see [11]) show that, the exponential dynamics of this reputation type works for all the parameters considered, and the master always reaches $p_{\mathcal{A}} = p_{\mathcal{A}}^{min}$ with eventual correctness. In addition, we see that the master also decreases its auditing cost, unlike the case of reputation type 1 where $p_{\mathcal{A}}$ goes to values close to one in order for the master to receive the correct reply with a high probability. We have also verified that when a majority (5 out of 9) of workers is covered, the system converges independently of the reputation type used.

Finally, for the sake of experimentation we checked that our mechanism reaches eventual correctness (with reputation type 2) by covering only 1 out of the 5 rational workers when the other 4 are malicious ones. The performance of the system to reach eventual correctness is similar to the analogous case where all workers are covered. Reputation type 1 and 3 have the same problems as before, whereas the fact that reputation becomes constant with type 2 allows rational covered workers to form a reputable set by itself and achieve fast eventual correctness.

Dynamic Change of Roles. As a further check of the stability of our procedure, we now study the case when after convergence is reached some workers change their type, possibly due to a software or hardware error. We simulate a situation in which 5 out of 9

rational workers suddenly change their behavior to malicious at time 500, a worst-case scenario. Figure 4 shows that after the rational behavior of 5 workers turns to malicious, convergence is reached again after a few hundred rounds and eventual correctness resumes. Notice that it takes more time for reputation type 2 to deal with the changes in the workers' behavior because this reputation can never increase, and hence the system will reach eventual correctness only when the reputation of the workers that turned malicious becomes less than the reputation of the workers that stayed rational. It also takes more time for reputation type 3 to deal with the changes in the worker' behavior. In the case of reputation type 1 not only the reputation of the workers that turned malicious decreases but also the reputation of the workers that stayed rational increases. Therefore, reputation type 1 exhibits better performance in dealing with dynamic changes of behavior than reputation types 2 and 3.

6 Conclusions and Future work

In this work we study a malicious-tolerant generic mechanism that uses reputation. We consider three reputation types, and give provable guarantees that only reputation type 2 (first presented here) provides eventual correctness in the case of covering only one altruistic or rational worker, something that is confirmed by our simulations. We show that reputation type 2 has more potential in commercial platforms where high reliability together with low auditing cost, rewarding few workers and fast convergence are required. This will help in developing reliable commercial Internet-based Master-Worker Computing services. From our simulations we make one more interesting observation: in the case when only rational workers exist and reputation type 3 (BOINC-like) is used, although the system takes more time to converge, in every round auditing is lower. Thus, reputation type 3 may fit better in volunteering setting where workers are most probably altruistic or rational and fast convergence can be sacrificed for lower auditing. In particular, our simulations reveal interesting tradeoffs between our reputation types and parameters and show that our mechanism is a generic one that can be adjusted to various settings. In a follow-up work we plan to investigate what happens if workers are connected to each other, forming a network (i.e, a social network through which they can communicate) or if malicious workers develop a more intelligent strategy against the system. Also the degree of trust among the players has to be considered and modeled in this scenario.

Acknowledgments. This work is supported by the Cyprus Research Promotion Foundation grant ΤΠΕ/ΠΛΗΡΟ/0609(ΒΕ)/05, the National Science Foundation (CCF-0937829, CCF-1114930), Kean University UFRI grant, Comunidad de Madrid grants S2009TIC-1692 and MODELICO-CM, and MICINN grants TEC2011-29688-C02-01 and PRODIEVO, and National Natural Science Foundation of China grant 61020106002.

References

1. Abraham, I., Dolev, D., Goden, R., Halpern, J.Y.: Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In: Proc. of PODC 2006, pp. 53–62 (2006)

2. Aiyer, A.S., Alvisi, L., Clement, A., Dahlin, M., Martin, J., Porth, C.: BAR fault tolerance for cooperative services. In: Proc. of SOSP 2005, pp. 45–58 (2005)
3. Amazon's Mechanical Turk, https://www.mturk.com
4. Anderson, D.: BOINC: A system for public-resource computing and storage. In: GRID (2004)
5. Anderson, D.: Volunteer computing: the ultimate cloud. Crossroads 16(3), 7–10 (2010)
6. BOINC reputation, http://boinc.berkeley.edu/trac/wiki/AdaptiveReplication
7. BOINC stats, http://boincstats.com/en/forum/10/4597
8. BOINC user survey, http://boinc.berkeley.edu/poll_results.php
9. Bush, R.R., Mosteller, F.: Stochastic Models for Learning. Wiley (1955)
10. Christoforou, E., Fernández Anta, A., Georgiou, C., Mosteiro, M., Sánchez, A.: Applying the dynamics of evolution to achieve reliability in master-worker computing. Concurrency and Computation: Practice and Experience (2013); A preliminary version appears in Euro-Par (2012)
11. Christoforou, E., Fernandez Anta, A., Georgiou, C., Mosteiro, M.A., Sánchez, A.: Reputation-based Mechanisms for Evolutionary Master-Worker Computing. ArXiv (2013)
12. The Einstein@home project, http://einstein.phys.uwm.edu
13. Estrada, T., Taufer, M., Anderson, D.P.: Performance prediction and analysis of BOINC projects: An empirical study with EmBOINC. J. of Grid Computing 7(4), 537–554 (2009)
14. Fernández, A., Georgiou, C., Lopez, L., Santos, A.: Reliable Internet-based computing in the presence of malicious workers. Parallel Processing Letters 22(1) (2012)
15. Fernández Anta, A., Georgiou, C., Mosteiro, M.A.: Designing mechanisms for reliable Internet-based computing. In: Proc. of NCA 2008, pp. 315–324 (2008)
16. Fernández Anta, A., Georgiou, C., Mosteiro, M.A.: Algorithmic Mechanisms for Internet-based Master-Worker Computing with Untrusted and Selfish Workers. In: IPDPS 2010 (2010)
17. Golle, P., Mironov, I.: Uncheatable distributed computations. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 425–440. Springer, Heidelberg (2001)
18. Heien, E.M., Anderson, D.P., Hagihara, K.: Computing low latency batches with unreliable workers in volunteer computing environments. J. of Grid Computing (2009)
19. Josang, A., Ismail, R., Boyd, C.: A Survey of Trust and Reputation Systems for Online Service Provision. Decision Support Systems Journal 43(2), 618–644 (2007)
20. Kondo, D., Araujo, F., Malecot, P., Domingues, P., Silva, L.M., Fedak, G., Cappello, F.: Characterizing result errors in internet desktop grids. In: Kermarrec, A.-M., Bougé, L., Priol, T. (eds.) Euro-Par 2007. LNCS, vol. 4641, pp. 361–371. Springer, Heidelberg (2007)
21. Konwar, K.M., Rajasekaran, S., Shvartsman, M.M.A.A.: Robust network supercomputing with malicious processes. In: Dolev, S. (ed.) DISC 2006. LNCS, vol. 4167, pp. 474–488. Springer, Heidelberg (2006)
22. Korpela, E., Werthimer, D., Anderson, D., Cobb, J., Lebofsky, M.: SETI@home: Massively distributed computing for SETI. Computing in Science and Engineering (2001)
23. Maynard-Smith, J.: Evolution and the Theory of Games. Cambridge University Press (1982)
24. Sarmenta, L.: Sabotage-tolerance mechanisms for volunteer computing systems. Future Generation Computer Systems 18(4), 561–572 (2002)
25. Shneidman, J., Parkes, D.C.: Rationality and self-interest in P2P networks. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, pp. 139–148. Springer, Heidelberg (2003)

26. Sonnek, J., Chandra, A., Weissman, J.B.: Adaptive Reputation-Based Scheduling on Unreliable Distributed Infrastructures. IEEE TPDS 18(11) (2007)
27. Szepesvári, C.: Algorithms for Reinforcement Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers (2010)
28. Taufer, M., Anderson, D., Cicotti, P., Brooks, C.L.: Homogeneous redundancy: a technique to ensure integrity of molecular simulation results using public computing. In: IPDPS (2005)
29. Vilaça, X., Denysyuk, O., Rodrigues, L.: Asynchrony and Collusion in the N-party BAR Transfer Problem. In: Even, G., Halldórsson, M.M. (eds.) SIROCCO 2012. LNCS, vol. 7355, pp. 183–194. Springer, Heidelberg (2012)
30. Yurkewych, M., Levine, B.N., Rosenberg, A.L.: On the cost-ineffectiveness of redundancy in commercial P2P computing. In: Proc. of CCS 2005, pp. 280–288 (2005)