CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 0000; **00**:1–18
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe

Applying the dynamics of evolution to achieve reliability in master-worker computing

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

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

SUMMARY

We consider Internet-based Master-Worker task computations, like SETI@home, where a master process sends tasks, across the Internet, to worker processes; workers execute, and report back some result. However, these workers are not trustworthy and it might be at their best interest to report incorrect results. In such master-worker computations, the behavior and the best interest of the workers might change over time.
We model such computations using evolutionary dynamics and we study the conditions under which the master can reliably obtain task results. In particular, we develop and analyze an algorithmic mechanism based on reinforcement learning to provide workers with the necessary incentives to eventually become truthful. Our analysis identifies the conditions under which truthful behavior can be ensured, and bounds the expected convergence time to that behavior. The analysis is complemented with illustrative simulations.
Copyright © 0000 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: Performing tasks, Internet-based computing, Evolutionary dynamics, Reinforcement learning, Algorithmic mechanism design.

1. INTRODUCTION

Motivation: As an alternative to expensive supercomputing parallel machines, Internet provides a feasible computational platform for processing complex computational jobs. Several Internet-based applications operate on top of this global computation infrastructure. Examples are volunteer-based "@home" projects [6] such as SETI [34] and profit-seeking computation platforms such as Amazon's Mechanical Turk [4].

Although the potential is great, the use of Internet-based computing is limited by the untrustworthy nature of the platform's components [6, 28, 30]. In SETI for example, there is a

*Correspondence to: Dept. of Computer Science, University of Cyprus, Nicosia, Cyprus. Email: chryssis@cs.ucy.ac.cy

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

Conference version: A preliminary version of this work, entitled "Achieving Reliability in Master-Worker Computing via Evolutionary Dynamics" appears in the proceedings of Euro-Par 2012, pages 451–463.

machine, call it the *master*, that sends tasks, across the Internet, to volunteers' computers, call them *workers*, that execute and report back some result. However, these workers may not be trustworthy and it might be at their best interest to report incorrect results; that is, workers (or their owners) can be viewed as *rational* [2,28,47]. In SETI, the master attempts to attenuate the impact of these bogus results by assigning the same task to several workers and comparing their outcomes (i.e., redundant task allocation is employed [6,46]).

Prior work [24, 25, 52] has shown that it is possible to design algorithmic mechanisms with reward/punishment schemes so that the master can reliably obtain correct task results. We view these mechanisms as one-shot in the following sense: In a round, the master sends a task to be computed to a collection of workers, and the mechanism, using auditing and reward/punishment schemes, guarantees (with high probability) that the master gets the correct task result. For another task to be computed, the process is repeated (with the same or different collection of workers) but without taking advantage of the knowledge gained.

Given a long running computation (such as SETI-like master-worker computations), it can be the case that the best interest, and hence the behavior of the workers, might change over time. The question then arises: Would it be possible to design a mechanism for performing many tasks, over the course of a possibly infinite computation, that could benefit from the repeated interaction between a master and the same collection of workers?

Our approach: In this work we provide a positive answer to the above question. To do so, we introduce the concept of *evolutionary dynamics* (widely used under the biological and social perspective) and apply it to Internet-based master-worker task computing. More specifically, we employ *reinforcement learning* [14,48] to model 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 likelihood of reusing the strategy just chosen, and negative payoffs reduce it. Payoffs are seen as parameterizations of players' responses to their experiences. Empirical evidence [11,15] suggests that reinforcement learning is more plausible with players that have information only on the payoffs they receive; i.e., they do not have knowledge of the strategies involved. This model of learning fits nicely in our master-worker computation problem: each worker have no information about the master and the other workers' strategies and it does not know the set of strategies that led to the payoff it receives. The workers have only information about the strategies they choose at each round and their own received payoffs. The master also has minimal information about the workers and their intentions (to be truthful or not). Thus, we employ reinforcement learning for both the master and the workers in an attempt to build a reliable computational platform.

Our contributions:

- To the best of our knowledge, this is the first work that studies the evolutionary dynamics of Internet-based master-worker task computing through reinforcement learning. We develop and analyze a mechanism based on reinforcement learning to be used by the master and the workers. In particular, in each round, the master allocates a task to the workers and decides whether to audit their responses with a certain probability $p_{\mathcal{A}}$. Depending on whether it audits or not, it applies a different reward/punishment scheme, and adjusts the probability $p_{\mathcal{A}}$ for the next round (also known as the next task execution). Similarly, in a round, each worker i decides, with a certain probability p_{Ci}, whether it will report an incorrect result or it will truthfully compute and report the correct task result. Depending on the outcome of its decision, measured by the increase or the decrease of the worker's *utility*, the worker adjusts its probability p_{Ci} for the next round.
- We show necessary and sufficient conditions under which the mechanism ensures *eventual correctness*. That is, we establish the conditions under which, after some finite number of rounds, the master obtains the correct task result in every round, with minimal auditing, while keeping the workers satisfied (with respect to their utility). Eventual correctness can be viewed as a form of *Evolutionary Stable Strategy* [19,27] as studied in Evolutionary Game Theory [50]: even if a "mutant" worker decides to change its strategy to cheating, it will soon be brought back to an honest strategy.

- Finally, we show that our mechanism, when adhering to the above-mentioned conditions, reaches eventual correctness quickly. In particular, we show analytically probabilistic bounds on the convergence time, as well as bounds on the expected convergence time. Our analysis is complemented with simulations, for a variety of parameter combinations likely to occur in practice.

Background and Related Work: An increasing number of works that apply game theory to distributed computing exists, including research on Internet routing, resource/facility location and sharing, containment of virus spreading, secret sharing, P2P services, and task computations. For more discussion on the connection between game theory and distributed computing we refer the reader to the surveys by Halpern [29] and by Abraham et al. [1], and to the book by Nisan et al. [41].

The problem of achieving reliability in master-worker computations has recently been studied under two different views: from a "classical" distributed computing view [23, 33, 46] and from a game-theoretic view [24, 52]. 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, such as being a wrongdoer intentionally. Altruistic workers exhibit a "good" behavior, that is, they always compute and return the correct task result. From the perspective of the master, the altruistic workers are the "correct" ones. Under this view, "classical" distributed computing models are defined (e.g., a fixed bound on the probability of a worker being malicious is assumed) and typical malicious-tolerant voting protocols are designed.

Under the classical game-theoretic view, workers act on their own *self-interest* and they do not have an a priori established behavior (malicious or altruistic). They are assumed to be *rational* [2, 28]. In other words, the workers decide on whether they will be *honest* and report the correct task result, or *cheat* and report a bogus result, depending on which strategy increases their benefit or *utility.* Under this view, Algorithmic Mechanisms [2, 17, 40] 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) [39], i.e., a strategy choice by each worker such that none of them has an incentive to change it. That Nash equilibrium is the one in which the master achieves a desired probability of obtaining the correct task result.

In a previous work by Fernández Anta et al. [24], a game-theoretic approach was used to achieve reliability in a similar Internet-based master-worker computation. A master processor assigns, across the Internet, a single computational task to a set of potentially untrusted worker processors and collects their responses. A weak form of collusion, as in this work, is assumed where all workers that cheat return the same incorrect result. Game-theoretic models for the problem were designed and analyzed, e.g., one master and one worker, several workers, etc. For each of those models, a mechanism that achieves high reliability under the same payoff system that we use was designed. In [24], besides rewarding the majority of workers, as we do here, other reward models where considered, e.g., rewarding all workers, rewarding none of the workers, etc. The work in [24] was later extended [25] to also consider the possibility of workers being malicious.

In a work by Yurkewych et al. [52] a similar game-theoretic approach is studied. The master can audit the results returned by rational workers with a tunable probability. Bounds for that audit probability are computed to guarantee that workers have incentives to be honest in three scenarios: redundant allocation with and without collusion, and single-worker allocation. They conclude that, in their model, single-worker allocation is a cost-effective mechanism, specially in presence of collusion. As opposed to [24], it is possible that a decision is not reached in a round, in which case a different set of workers is selected and the process is repeated (but without using any knowledge of the previous unsuccessful round).

The works in [24] and [52] construct a mechanism assuming a number of parameters are known to the master and the workers. In addition, the computation between master and workers is viewed as one-shot: each round of task executions is independent of the previous round. That is, their work does not consider the advantage of repeated communication with the same set of workers. Moreover it is expected that in a master-worker computation the behavior and the best interest of the workers change over time. A dynamic environment like that could be analyzed using evolutionary dynamics, originally introduced in biology as a tool to study the mathematical principles according to which life is evolving [42]. Since then, a number of fields were inspired by the principles of evolutionary dynamics (e.g., sociology, economics, artificial intelligence) and a variety of mechanisms was developed, aiming to accurately model the process of evolution. Our work is inspired by dynamics of evolution as a mean to model workers adaptation to a truthful behavior.

The dynamics of evolution have been studied under the principles of Evolutionary Game Theory (EGT). Maynard-Smith and Price [37,38] introduced the concept of EGT in an effort to apply game theoretical ideas to understand evolving populations of lifeforms. This made Maynard-Smith [37] adjust the traditional concept of strategy, equilibrium, and the nature of a player's interaction, so that a player would learn how to optimize its behavior and maximize its return. However, while in traditional Game Theory players choose a strategy from their strategy sets, EGT in biology is dealing with species inheriting possibly mutated strategies. When referring to social entities such as humans, evolution is understood as a learning process akin to "cultural evolution" [31]. Cultural evolution implies an analogy between learning and biological evolution. For the existing analogies between learning, at the individual level, and biological evolution, we refer the reader to a paper by Borgers and Sarin [13].

In EGT, instead of the Nash equilibrium, Maynard-Smith and Price used the Evolutionarily Stable Strategy (ESS) concept [19,27,45,50]. A strategy is called evolutionarily stable if, when the whole population is using this strategy, any group of invaders (mutants) using a different strategy will eventually die away over multiple generations (evolutionary rounds). All ESS are Nash equilibria but the reverse is not true. Our work is driven by the concept of ESS and we wish to have a similar stable strategy among workers that would guarantee reliability. Even if a mutant worker decides to change its strategy to cheating, it will be soon brought back to an honest strategy. Instead of the one-shot and repeated games of classical game theory, EGT assumes that the game is played repeatedly by players randomly drawn from large populations, uninformed of the preferences of opponents. In our work we do not wish to change the set of players as evolutionary rounds progress, but we rather talk about "cultural evolution", where workers change their strategies as a process of learning, rather than being replaced themselves.

While evolution operates on the global distribution of strategies within a given population, reinforcement learning [48] operates on the individual level of each member of the population. A well-known model of reinforcement learning is Bush and Mosteller's model [14]. In this model, the players have limited information and they play in discrete time repeatedly the same normal-form game. At each point in time, the players are characterized by a probability distribution over their strategy sets. Players' choices are random, since they are affected by some unpredictable "psychological" factor. This probability distribution is adjusted over time in response to experience. This experience is gained through repeated interactions of the players with the system, based on their strategies and the received payoffs. Positive payoffs reinforce the strategy just chosen, and negative payoffs discourage the use of that strategy.

Specifically, Bush and Mosteller's model is an aspiration-based reinforcement learning model: Players adapt by comparing their experience with an *aspiration* level. There are several models of how aspirations are formed and adjusted over time, formally described in a study by Bendor et al. [11]. In the present paper, we use a simple model where aspiration is fixed by the workers and does not change during the evolutionary process (as in [10]). For more information on the different reinforcement learning models and comparisons between them we refer the reader to a paper by Laslier et al. [35] and a study by Izquierdo and Izquierdo [32].

A survey by Phelps et al. [43] and an article by Conitzer and Sandholm [16] take a new approach on Mechanism Design by introducing the concept of Evolutionary Mechanism Design. Evolutionary mechanism design assumes an engineering approach, based on an incremental process that creates a partly automated mechanism design. The evolutionary mechanism has a continuous interaction and feedback from the current mechanism, as opposed to classical mechanism design, which after the mechanism is introduced in the system, remains in the same NE forever. Looking at it from a different perspective, evolutionary mechanism design is analogous to evolutionary game theory. Just as players may be forced to gradually adjust their strategies, in an analogous manner mechanisms are gradually making adjustments in their rules with respect to what strategies are currently in play. In some way, our mechanism can be seen as an evolutionary mechanism, since the probability of auditing of the master and the probability of cheating of the workers change, which is similar to changing the mechanism.

Distributed computation in the presence of selfishness was studied within the scope of combinatorial agencies in Economics [7–9, 20]. The basic model considered is a combinatorial variant of the classical principal-agent problem [36]: A master (principal) must motivate a collection of workers (agents) to exert costly effort on the master's behalf, but the workers' actions are hidden from the master. Instead of focusing on each worker's actions, the focus is on complex combinations of the efforts of the workers that influence the outcome. The principal-agent approach deals in general with designing contracts between the principal and the workers that allow the principal to get the most out of the workers without knowing a priori what their actual capabilities are. One difference with respect to our master-worker framework is that, the worker's actions cannot really be viewed as *hidden* in our setting. Another important difference is that our scheme considers worker punishment, as opposed to the schemes in combinatorial agency where workers cannot be fined (limited liability constraint); this is possible in our framework as worker's actions are contractible (either a worker truthfully performs a task or not).

In the work of Rose and Willemain [44] the principal-agent problem is extended to evolutionary learning, and bounded rationality of the agents is assumed. Players' learning is simulated with a genetic algorithm that roughly mimics selection and mutations in biological evolution. Changes in the system are externally induced through the use of incentives. The agents' learning is aided by the principal's incentives that are used to adjust the learning, according to the output the principal desires. The principal is also able to use an artificial selection procedure to identify high performing agents for its own benefit.

Compared with the work of Rose and Willemain [44], in our line of work the learning model is different (in addition to the differences our work has with the principal-agent model). We assume that the learning procedure of the players remains the same through the evolutionary process. In contrast, in the more general model of Rose and Willemain, the learning procedure of the players may change over time and players can experience mutations. In both works incentives are used to impose a desired behavior over time. But in [44] bounded rationality has been used, while in our work no cognitive limitation of the workers is assumed. As a future direction, our model could be enriched by including 'biological learning' through replicator dynamics. While in [44] the principal artificially selects the agents that increase performance, in our context the master could, by using a reputation technique, exclude from the computation low performing workers.

2. MODEL AND DEFINITIONS

Master-Worker Framework: We consider a distributed system consisting of a master processor that assigns, over the Internet, computational tasks to a set W of n workers (with out loss of generality, we assume that n is odd). In particular, the computation is broken into rounds. In each round the master sends a task to be computed to the workers and the workers return the task result. The master, based on the workers' replies, must decide on the value it believes is the correct outcome of the task in the same round. The tasks considered in this work are assumed to have a unique solution; although such limitation reduces the scope of application of the presented mechanism [49], there are plenty of computations where the correct solution is unique: e.g., any mathematical function. Note that in this work we do not focus on any specific application. Therefore, applying our mechanism in practice may require to pay attention to details depending on the specific application considered. In this work security issues are not considered. Security can be achieved by cryptographic means, as done in BOINC [3], which allows for encrypting communication, authenticating master and workers, signing the code of tasks and executing tasks in sandboxes.

Following Abraham et al. [2], and Shneidman and Parkes [47], we assume that workers are *rational*, that is, they are selfish in a game-theoretic sense and their aim is to maximize their benefit (utility) under the assumption that other workers do the same. In the context of this paper, a worker is *honest* in a round when it truthfully computes and returns the task result, and it *cheats* when it returns some incorrect value. So, a worker decides to be honest or to cheat depending on which strategy maximizes its utility. We denote by p_{Ci}^{r} the probability of a worker i cheating in round r. This probability is not fixed, as the worker adjusts it over the course of the computation.

While it is assumed that workers make their decision individually and with no coordination, it is assumed that all the workers that cheat in a round return the same incorrect value (as done, for example, in [23, 24, 46]). This assumption yields a worst case scenario (and hence analysis) for the master with respect to obtaining the correct result; it subsumes models where cheaters do not necessarily return the same answer. (In some sense, this can be seen as a cost-free, weak form of collusion.)

Auditing, Payoffs, Rewards and Aspiration: To "persuade" workers to be honest, the master employs, when necessary, *auditing* and *reward/punishment* 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 $p_{\mathcal{A}}$ the probability of the master auditing the responses of the workers. The master can change this auditing probability over the course of the computation. Unless otherwise stated, we assume that there is a value $p_{\mathcal{A}}^{min} > 0$ so that at all times $p_{\mathcal{A}} \geq p_{\mathcal{A}}^{min}$. Furthermore, the master can reward and punish workers, which can be used (possibly combined with auditing) to encourage workers to be honest. When the master audits, it can accurately reward and punish workers. When the master does not audit, it decides on the majority of the received replies, and it rewards only the majority. We refer to this as the $\mathcal{R}_{\mathrm{m}}$ reward scheme (as presented in previous works).

The payoff parameters considered in this work are detailed in Table I. Note that the first letter of the parameter's name identifies whose parameter it is. M stands for master and W for worker. Then, the second letter gives the type of parameter. P stands for punishment, C for cost, and B for benefit. 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.

$WP_{\mathcal{C}}$	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 I. Payoffs. The parameters are non-negative.

We assume that each worker i has an aspiration a_i (the same in all rounds) which is the minimum benefit it expects to obtain in a round. In order to motivate the worker to participate in the computation, the master must ensure 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. This information can be included, for example, in a contract the master and the worker agree upon, prior to the start of the computation.

Among the parameters involved, we assume that the master has the freedom of choosing $WB_{\mathcal{Y}}$ and $WP_{\mathcal{C}}$; by tuning these parameters and choosing n, the master tries to achieve the goal of eventual correctness (see below). All other parameters can either be fixed because they are system parameters, or may also be chosen by the master (except the aspiration, which is a parameter set by each worker).

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

Algorithm 1 Master's Algorithm

```
$p_{\mathcal{A}} \leftarrow x$, where $x \in [p_{\mathcal{A}}^{min}, 1]$
for $r \leftarrow 1$ to $\infty$ 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
      accept the majority
    else
      $p'_{\mathcal{A}} \leftarrow p_{\mathcal{A}} + \alpha_m(cheaters(r)/n - \tau)$
      $p_{\mathcal{A}} \leftarrow \min\{1, \max\{p_{\mathcal{A}}^{min}, p'_{\mathcal{A}}\}\}$
    $\forall i \in W$ : pay/charge $\Pi_i$ to worker $i$
```

Algorithm 2 Algorithm for Worker i

```
$p_{Ci} \leftarrow y$, where $y \in [0,1]$
for $r \leftarrow 1$ to $\infty$ do
  receive a task $T$ from the master
  set $S_i \leftarrow -1$ 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 $\sigma$ to the master
  get payoff $\Pi_i$
  $p'_{Ci} \leftarrow p_{Ci} - \alpha_w(\Pi_i - a_i)S_i$
  $p_{Ci} \leftarrow \max\{0, \min\{1, p'_{Ci}\}\}$
```

3. ALGORITHMIC MECHANISM

We now detail the algorithms run by the Master and the workers.

Master's Algorithm (Alg. 1): The master's algorithm begins by choosing the initial probability of auditing. After that, at each round, the master sends a task to all workers and, after all answers are received (a reliable network is assumed), the master audits the answers with probability $p_{\mathcal{A}}$. In the case the answers are not audited, the master accepts the value contained in the majority of answers and continues to the next round with the same probability of auditing. In the case the answers are audited, the value $p_{\mathcal{A}}$ of the next round is reinforced (i.e., modified according to the outcome of the round). Then, the master rewards/penalizes the workers accordingly.

The master initially has scarce or no information about the environment (e.g., workers initial p_C). The initial probability of auditing will be set according to the information the master possesses. For example, if it has no information about the environment, a safe approach may be to initially set $p_{\mathcal{A}} = 0.5$.

Observe that, when the answers are not audited, the master has no information about the number of cheaters in the round. Thus, the probability $p_{\mathcal{A}}$ remains the same as in the previous round. When the answers are audited, the master can determine the number of cheaters; we denote by $cheaters(r)$ the number of cheaters in round r. Then, the master adapts the auditing probability $p_{\mathcal{A}}$ according to this number. Observe that the algorithm guarantees $p_{\mathcal{A}} \geq p_{\mathcal{A}}^{min}$. This, combined with the property $p_{\mathcal{A}}^{min} > 0$, will prevent the system to fall in a permanent set of "bad" states where $p_{\mathcal{A}} = 0$ and $p_C > 0$. A discount factor, which we call *tolerance* and denote by τ, expresses the master's tolerable ratio of cheaters (typically, we will assume $\tau = 1/2$). Hence, if the proportion of cheaters is larger than τ, $p_{\mathcal{A}}$ will be increased, and otherwise, $p_{\mathcal{A}}$ will be decreased. The amount by which $p_{\mathcal{A}}$ changes depends on the change in the number of cheaters, 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 $p_{\mathcal{A}}$.)

Workers' Algorithm (Alg. 2): The workers' algorithm begins with each worker i deciding an initial probability of cheating p_{Ci}. At each round, each worker receives a task from the master and, with probability $1 - p_{Ci}$ calculates the task, and replies to the master with the correct answer. If the worker decides to cheat, it fabricates an answer and sends the incorrect response to the master. (We use a flag S_i to model the decision of a worker i to cheat or not.) After receiving its payoff (detailed in the analysis section), each worker i changes its p_{Ci} according to the payoff Π_i received, the chosen strategy S_i, and its aspiration a_i. Observe that the workers' algorithm guarantees $0 \leq p_{Ci} \leq 1$.

Workers have a learning rate α_w. We assume that all workers have the same learning rate, that is, they learn in the same manner (see the discussion in [48]; the learning rate is called step-size there); note that our analysis can be adjusted to accommodate also workers with different learning rates. We choose the value of α_w so that $\alpha_w(a_i + WP_{\mathcal{C}}) < 1, \ \forall i \in W$. Otherwise, the system could enter in an oscillating condition where some nodes alternate p_C between 0 and 1 never converging to a stable state, which is necessary to guarantee reliability.

4. ANALYSIS

In this section we analyze the mechanism presented in Section 3. We model the evolution of the mechanism as a Markov chain, and we prove necessary and sufficient conditions for achieving eventual correctness. We provide analytical evidence that convergence to eventual correctness can be reached rather quickly. Observe in Algorithms 1 and 2 that there are a number of variables that may change in each round. We will denote the value of a variable X after a round r with a superindex r, as X^r.

4.1. The Mechanism as a Markov Chain

We analyze the evolution of the master-workers system as a Markov chain. To do so, we first define the set of states and the transition function as follows.

Let the state of the Markov chain be given by the vector of probabilities $(p_{\mathcal{A}}, p_{C1}, p_{C2}, \ldots, p_{Cn})$. Then, we denote the state after round r by $(p_{\mathcal{A}}^r, p_{C1}^r, p_{C2}^r, \ldots, p_{Cn}^r)$. Observe from Algorithms 1 and 2 that any state $(p_{\mathcal{A}}, p_{C1}, p_{C2}, \ldots, p_{Cn})$ in which $p_{\mathcal{A}} \in [p_{\mathcal{A}}^{min}, 1]$ and $p_{Ci} \in [0, 1]$ for each worker i, is a possible initial state of the Markov chain. The workers' decisions, the number of cheaters, and the payoffs in round r are the stochastic outcome of the probabilities used in round r. Then, restricted to $p_{\mathcal{A}}^r \in [p_{\mathcal{A}}^{min}, 1]$ and $p_{Ci}^r \in [0, 1]$, we can describe the transition function of the Markov chain in detail. For each subset of workers $F \subseteq W$, $P(F) = \prod_{j \in F} p_{Cj}^{r-1} \prod_{k \notin F} (1 - p_{Ck}^{r-1})$ is the probability that the set of cheaters is exactly F in round r. Then, we have the following.

- With probability $p_{\mathcal{A}}^{r-1} \cdot P(F)$, the master audits when the set of cheaters is F, and then,
  (0) the master updates $p_{\mathcal{A}}$ as $p_{\mathcal{A}}^r = p_{\mathcal{A}}^{r-1} + \alpha_m(|F|/n - \tau)$, and
  (1) each worker $i \in F$ updates p_{Ci} as $p_{Ci}^r = p_{Ci}^{r-1} - \alpha_w(a_i + WP_{\mathcal{C}})$,
  (2) each worker $i \notin F$ updates p_{Ci} as $p_{Ci}^r = p_{Ci}^{r-1} + \alpha_w(a_i - (WB_{\mathcal{Y}} - WC_{\mathcal{T}}))$.
- With probability $(1 - p_{\mathcal{A}}^{r-1})P(F)$, the master does not audit when F is the set of cheaters. Then, the master does not change $p_{\mathcal{A}}$ and the workers update p_{Ci} as follows. For each $i \in F$,
  (3) if $|F| > n/2$ then $p_{Ci}^r = p_{Ci}^{r-1} + \alpha_w(WB_{\mathcal{Y}} - a_i)$,
  (4) if $|F| < n/2$ then $p_{Ci}^r = p_{Ci}^{r-1} - \alpha_w \cdot a_i$,

  and for each $i \notin F$,
  (5) if $|F| > n/2$ then $p_{Ci}^r = p_{Ci}^{r-1} + \alpha_w(a_i + WC_{\mathcal{T}})$,
  (6) if $|F| < n/2$ then $p_{Ci}^r = p_{Ci}^{r-1} + \alpha_w(a_i - (WB_{\mathcal{Y}} - WC_{\mathcal{T}}))$.

The following terminology will be used throughout. Let a *covered worker* be one that is paid at least its aspiration a_i and the computing cost $WC_{\mathcal{T}}$. In any given round r, let an *honest worker* be one for which $p_C^{r-1} = 0$. Let an *honest state* be one where the *majority* of workers are honest. Let an *honest set* be any set of honest states. We refer to the opposite cases as *uncovered worker*, *cheater worker* ($p_C^{r-1} = 1$), *cheat state*, and *cheat set* respectively.

4.2. Conditions for Eventual Correctness

We show the conditions under which the system can guarantee eventual correctness. We begin with some terminology. 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 order to show eventual correctness, we must show eventual convergence to a closed honest set. Thus, we need to show (i) that there exists at least one such closed honest set, (ii) that all closed sets are honest, and (iii) that one honest closed set is reachable from any initial state. Lemma 1 shows that, if $p_A = 0$ then some cheat set is closed. Given (ii), the necessity of $p_{\mathcal{A}}^{min} > 0$ is motivated by this claim. Hence, $p_A > 0$ is assumed for the rest of the analysis. Lemma 2 shows that, if the majority of workers is uncovered, no honest set is closed. Given (i), the necessity of a covered majority is motivated. Hence, it is assumed that the majority of workers are covered for the rest of the analysis.

Lemma 3 shows that the honest set including all the states in which all covered workers are honest is closed, which proves (i). Lemma 4 shows that any honest set where some covered worker is not honest is not closed, and Lemma 5 shows that any set that is not honest is not closed. Together, they prove (ii), and also (iii) because, if only honest sets are closed, there is a way of going from non-honest sets to one of them. The overall result is established in Theorem 6.

Lemma 1
Consider any set of workers $Z \subseteq W$ such that $\forall i \in Z : WB_{\mathcal{Y}} \geq a_i$. If $|Z| > n/2$, then the set of states $S = \{(p_{\mathcal{A}}, p_{C1}, \ldots, p_{Cn}) | (p_{\mathcal{A}} = 0) \wedge (\forall w \in Z : p_{Cw} = 1)\}$, is a closed cheat set.

Proof
Observe first that each state in S is a cheat state, since the master does not audit and a majority of workers cheat. From transition (3) it can be seen that, if the chain is in a state of the set S before round r, for each worker $i \in Z$, $p_{Ci}^{r} \geq p_{Ci}^{r-1} = 1$ holds. In addition, $p_{\mathcal{A}}$ does not change. Hence, once the chain has reached a state in the set S, it will move only to states in the set S. □

As already mentioned, from this lemma, it is concluded that $p_{\mathcal{A}} > 0$ is required for eventual correctness. From now on, it is assumed that in all rounds $p_{\mathcal{A}} \geq p_{\mathcal{A}}^{min} > 0$.

Lemma 2
If there exists a set of workers $Z \subseteq W$ such that $|Z| > n/2$ and $\forall i \in Z : WB_{\mathcal{Y}} < a_i + WC_{\mathcal{T}}$ then no honest set is closed.

Proof
Recall that we choose the value of α_w so that $\forall i \in W : \alpha_w(a_i + WP_{\mathcal{C}}) < 1$. Consider any starting state, which by assumption is an honest state, $S = \{(p_{\mathcal{A}}, p_{C1}, \ldots, p_{Cn}) | \exists Y \subseteq W : (|Y| > n/2) \wedge (\forall w \in Y : p_{Cw} = 0)\}$.

Let the set Z be divided in three sets depending on whether the workers are honest ($p_C = 0$), cheaters ($p_C = 1$) or cheat with a probability between zero and one ($0 < p_C < 1$). We denote these sets by Z_0, Z_1 and Z_b respectively. In the next round the master audits (possible since $p_{\mathcal{A}} > 0$), workers in Z_0 and Z_b do not cheat and workers in Z_1 cheat. Then from transition (2) all workers in Z_0 and Z_b increase their probability of cheating. From transition (1) all workers in Z_1 decrease their cheating probability by $\alpha_w(a_i + WP_{\mathcal{C}})$. Since all workers in Z_1 are cheater workers ($p_C = 1$) and $\alpha_w(a_i + WP_{\mathcal{C}}) < 1$, after this round their cheating probability is larger than 0. Hence, for all workers in Z their cheating probability is larger than 0 and the new state is not honest. □

Lemma 3
Consider any set of workers $Z \subseteq W$ such that $\forall i \in Z : WB_{\mathcal{Y}} \geq a_i + WC_{\mathcal{T}}$ and $\forall j \notin Z : WB_{\mathcal{Y}} < a_j + WC_{\mathcal{T}}$. If $|Z| > n/2$, then the set of states $S = \{(p_{\mathcal{A}}, p_{C1}, \ldots, p_{Cn}) | \forall w \in Z : p_{Cw} = 0\}$, is an honest closed set.

Proof
Consider any round r before which the state of the chain is $s \in S$. Given that $|Z| > n/2$, at round r we have $cheaters(r) < n/2$. Then, for all workers in Z, the transition function is either (2) or (6), depending on whether the master audits or not. Then, given that $WB_{\mathcal{Y}} \geq a_i + WC_{\mathcal{T}}$ for all workers in Z, their probability of cheating after round r is still 0. Hence, the claim follows. □

Lemma 4
Consider any set of workers $Z \subseteq W$ such that $\forall i \in Z : WB_{\mathcal{Y}} \geq a_i + WC_{\mathcal{T}}$ and $\forall j \notin Z : WB_{\mathcal{Y}} < a_j + WC_{\mathcal{T}}$. Then, for any set of states $S = \{(p_{\mathcal{A}}, p_{C1}, \ldots, p_{Cn}) | \exists Y \subseteq W : (|Y| > n/2) \wedge (\forall w \in Y : p_{Cw} = 0) \wedge (Z \nsubseteq Y)\}$, S is not a closed set.

Proof
For the sake of contradiction assume that S is a closed set. Then, after a round r when the state is $s \in S$, the chain remains in S forever. Given that $|Y| > n/2$ and the assumption that S is a closed set, at all rounds $r' > r$ we must have $cheaters(r') < n/2$. Then, given that $\forall i \in Z : WB_{\mathcal{Y}} \geq a_i + WC_{\mathcal{T}}$, from the transition function it can be seen that the probability of cheating of all workers in Z

decreases in every round, independently of whether the master audits or not. But then, at some round $r' > r$, for all $i \in Z$, $p_{Ci}^{r'} = 0$ must hold. Then, $Z \subseteq Y$ at round r', showing that S is not closed. □

Lemma 5
Consider any set of workers $Z \subseteq W$ such that $\forall i \in Z : WB_{\mathcal{Y}} \geq a_i + WC_{\mathcal{T}}$ and $\forall j \notin Z : WB_{\mathcal{Y}} < a_j + WC_{\mathcal{T}}$. If $|Z| > n/2$ and $p_{\mathcal{A}} > 0$, then for any set of states $S = \{(p_{\mathcal{A}}, p_{C1}, \ldots, p_{Cn}) | \exists Y \subseteq W : (|Y| > n/2) \wedge (\forall w \in Y : p_{Cw} > 0)\}$, S is not a closed set.

Proof
To prove this claim, it is enough to show that, after a round r when the state is $s \in S$, with some positive probability the chain moves out of S. Assume that, starting at some round $r' \geq r$, the master audits in all rounds in $[r', r'']$, for a suitable r''. Such assumption is valid because $p_{\mathcal{A}} > 0$. Then, given that $\forall i \in Z : WB_{\mathcal{Y}} \geq a_i + WC_{\mathcal{T}}$, from the transition function it can be seen that the probability of cheating of all workers in Z decreases in every round by an amount not smaller than $\alpha_w \min\{WB_{\mathcal{Y}} - a_i - WC_{\mathcal{T}}, WP_{\mathcal{C}} + a_i\}$. But then, at some round $r'' > r$, for all $i \in Z$ we have $p_{Ci}^{r''} = 0$. Therefore, $|Y| < n/2$ at round r'' showing that S is not closed. □

The following theorem shows that there is a positive probability of reaching some state after which correctness can be guaranteed, as long as for a chosen majority of workers, the payment is enough to cover their aspiration and cost of performing the task. Its proof follows directly from Lemmas 3– 5.

Theorem 6
If $p_{\mathcal{A}} > 0$ and for all $i \in W : \alpha_w(a_i + WP_{\mathcal{C}}) < 1$ then, in order to guarantee with positive probability that, after some finite number of rounds, the system achieves eventual correctness, it is **necessary** and **sufficient** to set $\boxed{WB_{\mathcal{Y}} \geq a_i + WC_{\mathcal{T}}}$ for all $i \in Z$ in some set $Z \subseteq W$ such that $|Z| > n/2$.

Remark: From Algorithm 1 it is easy to see that once the closed set $S = \{(p_{\mathcal{A}}, p_{C1}, \ldots, p_{Cn}) | \forall w \in Z : p_{Cw} = 0\}$ is reached, eventually $p_{\mathcal{A}} = p_{\mathcal{A}}^{min}$ and stays such forever.

4.3. Convergence Time

Theorem 6 gives necessary and sufficient conditions to achieve eventual correctness. However, in order to have a practical system, it is necessary to bound the time taken to achieve it, which we call the *convergence time*. In other words, starting from any initial state, we want to compute the number of rounds that the Markov chain takes to reach an honest closed set. In this section, we show bounds on the convergence time.

Expected Convergence Time: Let C be the set of all covered workers. We assume, as required by Theorem 6, that $|C| > n/2$. From transitions (1) and (2) in the Markov chain definition, it can be seen that it is enough to have a consecutive sequence of $1/(\alpha_w \min\{WB_{\mathcal{Y}} - a_i - WC_{\mathcal{T}}, WP_{\mathcal{C}} + a_i\})$ audits to enforce $p_C = 0$ for all covered workers $i \in C$. This gives the following upper bound on the convergence time:

Theorem 7
The expected convergence time is at most $\rho/(p_A^{min})^{\rho}$, where $\rho = 1/(\alpha_w \min_{i \in C}\{WB_{\mathcal{Y}} - a_i - WC_{\mathcal{T}}, WP_{\mathcal{C}} + a_i\})$ and C is the set of covered workers.

Proof
The expected convergence time is upper bounded by the expected time for ρ consecutive audits. Consider the time divided in phases of ρ rounds. Let a phase where the master audits in all rounds be called *successful*. The expected time for ρ consecutive audits is at most the expected time for a successful phase. The probability of success in any given phase is at least $(p_A^{min})^{\rho}$. Consider the probability distribution of the number X of phases needed to have success, each with probability $(p_A^{min})^{\rho}$. This distribution is geometric and the expectation of X is $1/(p_A^{min})^{\rho}$. Given that each phase has ρ rounds, the claim follows. □

The upper bound shown in Theorem 7 may be too pessimistic for certain values of the parameters. The following theorem provides a tighter bound under certain conditions.

Theorem 8
Let us define, for each worker i, $dec_i \triangleq \alpha_w \min\{WP_{\mathcal{C}} + a_i, WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_i\}, inc_i \triangleq \alpha_w \max\{WB_{\mathcal{Y}} - a_i, WC_{\mathcal{T}} + a_i\}$. Let C be the set of covered workers. If $p_{\mathcal{A}}^{min} = \max_{i\in C}\{inc_i/(inc_i + dec_i)\} + \varepsilon$, for some $0 < \varepsilon < 1 - \max_{i\in C}\{inc_i/(inc_i + dec_i)\}$, then the expected convergence time is $1/(\varepsilon(\min_{i\in C}\{dec_i\} + \max_{i\in C}\{inc_i\}))$.

Proof
Let us define a potential function ϕ over the rounds as follows. Initially $\phi(0) = \max_{i\in C}\{p_{\mathcal{C}i}^0\}$. Then, for each round $r > 0$, $\phi(r) = \phi(r-1)$ if $\phi(r-1) = 0$. If $\phi(r-1) > 0$, then $\phi(r) = \max(0, \phi(r-1) - \min_{j\in C}\{dec_j\})$ if the master audits in round $r > 0$, and $\phi(r) = \phi(r-1) + \max_{j\in C}\{inc_j\}$ otherwise.

Consider any worker $i \in C$, and observe that, in the extreme cases, $p_{\mathcal{C}i}$ decreases by $\min_{j\in C}\{dec_j\}$ when the master audits and increases by $\max_{j\in C}\{inc_j\}$ when the master does not audit. Also, once all workers in C have $p_{\mathcal{C}i} = 0$, this value does not change, since there is a majority of honest workers. Hence, it is clear that for all $r \geq 0$, $\phi(r) \leq \max_{i\in C}\{p_{\mathcal{C}i}^r\}$.

As a worst case, assume that $\phi(0) = 1$. We compute the expected number of rounds needed to get $\phi = 0$ as follows. We need $1/dec_i$ audits for each $1/inc_i$ non-audit rounds to compensate for the increase in potential. Setting $p_{\mathcal{A}}^{min} = inc_i/(inc_i + dec_i)$ the master achieves at least that ratio in expectation for any time period. (Omitting that time is discrete for clarity.) Additionally, in order to compensate for the initial $\phi(0) = 1$, $1/dec_i$ additional audits are needed. Making $p_{\mathcal{A}}^{min} = inc_i/(inc_i + dec_i) + \varepsilon$, for some $0 < \varepsilon < 1 - inc_i/(inc_i + dec_i)$, the expected convergence time is $1/(\varepsilon(\min_{i\in C}\{dec_i\} + \max_{i\in C}\{inc_i\}))$. $\square$

The following corollary is derived from the previous theorem for a suitable scenario.

Corollary 9
If $WP_{\mathcal{C}} + a_i \geq WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_i$ and $WB_{\mathcal{Y}} - a_i \leq WC_{\mathcal{T}} + a_i$, $\forall i \in C$, and if $p_{\mathcal{A}}^{min} = \frac{WC_{\mathcal{T}} + \max_{i\in C} a_i}{WB_{\mathcal{Y}}} + \varepsilon$, where C is the set of covered workers and $0 < \varepsilon < 1 - (WC_{\mathcal{T}} + \max_{i\in C} a_i)/WB_{\mathcal{Y}}$, then the expected convergence time is ρ/ε, where $\rho = 1/(\alpha_w WB_{\mathcal{Y}})$.

Probabilistic Bound on the Number of Rounds for Convergence: We show now that, under certain conditions on the parameters of the system, it is possible to bound the probability to achieve convergence and the number of rounds to do so. Assume that $p_{\mathcal{A}}^0 > 0$. Since $p_{\mathcal{A}}$ is not changed unless the master audits, we have the following:

Lemma 10
Let $p_{\mathcal{A}}^0 = p > 0$. Then, the master audits in the first $\rho = \ln(1/\varepsilon_1)/p$ rounds with probability at least $1 - \varepsilon_1$, for any $\varepsilon_1 \in (0, 1)$.

Proof
The master audits in the first ρ rounds with probability $1 - (1-p)^\rho \geq 1 - \exp(-\rho \cdot p) = 1 - \varepsilon_1$. $\square$

Let us assume that the system parameters are such that, for all workers i, $\alpha_w(WP_{\mathcal{C}} + a_i) \in [0, 1]$ and $\alpha_w(WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_i) \in (0, 1]$ (all workers are covered). Let us define $dec\_cheater \triangleq \alpha_w \min_i\{WP_{\mathcal{C}} + a_i\}$ and $dec\_honest \triangleq \alpha_w \min_i\{WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_i\}$. From transitions (1) and (2) we derive the following lemma:

Lemma 11
Let r be a round in which the master audits, and F be the set of cheaters in round r. Then,

$$p_{\mathcal{C}i}^r \leq 1 - \alpha_w(WP_{\mathcal{C}} + a_i) \leq 1 - dec\_cheater, \forall i \in F$$

$$p_{\mathcal{C}j}^r \leq 1 - \alpha_w(WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_j) \leq 1 - dec\_honest, \forall j \notin F$$

Let us denote the sum of all cheating probabilities before a round r as $P^{r-1} \triangleq \sum_i p_{Ci}^{r-1}$.

Lemma 12
Let r be a round in which the master audits such that $P^{r-1} > n/3$. If $dec\_cheater \geq dec\_honest$ and $dec\_cheater + 3 \cdot dec\_honest \geq 8/3$, then $P^r \leq n/3$ with probability at least $1 - \exp(-n/96)$.

Proof
Let F be the set of cheaters in round r. Then, using a Chernoff bound $Pr[|F| < (1-\delta)P^{r-1}] \leq \exp(-\delta^2 P^{r-1}/2)$, for any $\delta \in (0,1)$. Then, since $P^{r-1} > n/3$, using $\delta = 1/4$, there are at least $(1-\delta)P^{r-1} > n/4$ cheaters with probability at least $1 - \exp(-\delta^2 P^{r-1}/2) > 1 - \exp(-n/96)$. If that is the case, from Lemma 11 and $dec\_cheater \geq dec\_honest$, we have that

$$\begin{aligned} P^r &\leq n - |F| dec\_cheater - (n - |F|) dec\_honest \leq n - (n/4) dec\_cheater - (3n/4) dec\_honest \\ &= n(1 - (dec\_cheater + 3 \cdot dec\_honest)/4) \leq n/3, \end{aligned}$$

as desired. □

Let us now define $dec_i \triangleq \alpha_w \min\{a_i, WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_i\}$. Let, $dec \triangleq \min_i dec_i$. Assume $WP_{\mathcal{C}} \geq 0$ and $a_i \geq 0$, for all workers.

Lemma 13
Consider a round r such that $P^{r-1} \leq n/3$. Then, with probability at least $1 - \exp(-n/36)$ each worker i has $p_{Ci}^r \leq \max\{0, p_{Ci}^{r-1} - dec\}$, and hence $P^r \leq n/3$.

Proof
Using Chernoff, there is a majority of honest workers with probability at least

$$Pr[majority\ honest | P^{r-1} \leq n/3] \geq 1 - \exp(-(1/2)^2(n/3)/3) = 1 - \exp(-n/36).$$

It can be observed in Algorithm 2 that, if there is a majority of honest workers in a round r, then any worker i has $p_{Ci}^r \leq \max\{0, p_{Ci}^{r-1} - dec\}$, independently of whether the master audits. Hence the proof. □

Theorem 14
Assume $\alpha_w(WP_{\mathcal{C}} + a_i) \in [0,1]$ and $\alpha_w(WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_i) \in (0,1]$ for all workers i. (Observe that all workers are covered.) Let $dec\_cheater \triangleq \alpha_w \min_i\{WP_{\mathcal{C}} + a_i\}$, $dec\_honest \triangleq \alpha_w \min_i\{WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_i\}$, and $dec \triangleq \alpha_w \min_i\{a_i, WB_{\mathcal{Y}} - WC_{\mathcal{T}} - a_i\}$. If $p_{\mathcal{A}}^0 = p > 0$, $dec\_cheater \geq dec\_honest$ and $dec\_cheater + 3 \cdot dec\_honest \geq 8/3$, then eventual convergence is reached in at most $\ln(1/\varepsilon_1)/p + 1/dec$ rounds, with probability at least $(1-\varepsilon_1)(1-\exp(-n/96))(1-\exp(-n/36))^{1/dec}$, for any $\varepsilon_1 \in (0,1)$.

Proof
Consider the first round r in which the masters audits. From Lemma 10, r is in the first $\ln(1/\varepsilon_1)/p$ rounds with probability at least $1 - \varepsilon_1$. If so, either $P^{r-1} \leq n/3$ and also $P^r \leq n/3$ (from Algorithm 2 and the fact that the master audits in round r, P cannot increase in round r), or $P^{r-1} > n/3$. In this latter case, from Lemma 12, $P^r \leq n/3$ with probability at least $1 - \exp(-n/96)$. Then starting at round $r+1$, from Lemma 13, with probability at least $(1 - \exp(-n/36))^{1/dec}$, there are $1/dec$ consecutive rounds with majorities of honest workers. Since in each of these rounds the cheating probability of any worker decreases at least by dec (unless it is already zero), at the end of these rounds all workers have zero cheating probability. □

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, a concept hard to view through the analysis. This is important as our analytical upper bounds on convergence time correspond to worst case scenarios. Here, we present simulations for a variety of parameter

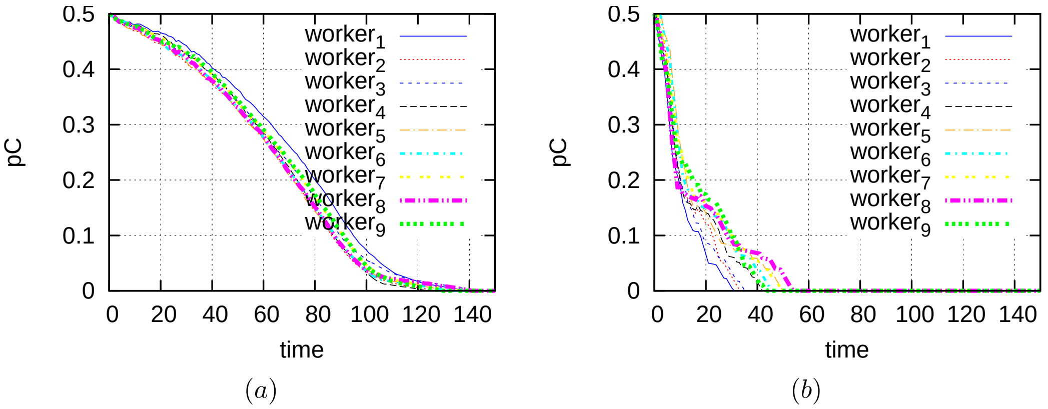

Figure 1. Cheating probability for each worker as a function of time (number of rounds) for parameters $p_C = p_{\mathcal{A}} = 0.5$, $WB_{\mathcal{Y}} = 1$, $WP_{\mathcal{C}} = 0$, $WC_{\mathcal{T}} = 0.1$ and $a_i = 0.1$. (a) $\alpha = 0.01$; (b) $\alpha = 0.1$.

combinations likely to occur in practice. We have created our own simulation setup by implementing our mechanism (the master's and the workers' algorithms) using the C++ programming language. We have run our simulations on a PC with an Intel Core 2 Duo, 2.80GHz CPU, 4GB of RAM and Ubuntu 11.04 OS. Each depicted plot value represents the average over 10 executions of the implementation. The plots of Figure 4 are an exception and the plotted values represent only 1 execution of the implementation; the purpose is to illustrate the per round cost of the master. We have simulated several scenarios for different parameter values (here we present selected results).

Simulation parameters: We choose sensible parameter values likely to be encountered in real applications; the choice of the parameters was influenced by statistics obtained from experiments contacted in SETI-like projects ([21, 22, 51]). In particular, the number of workers has been set to nine (an odd number to accommodate majority voting when the master does not audit). In systems like Seti@home typically each task is assigned to three workers [34]. So, in that context, nine workers seems an appropriate workforce. The initial cheating probability of each worker i is not known, therefore we have experimented with $p_{Ci} = 0.5$, as a reasonable assumption, and with $p_{Ci} = 1$ as an extreme case. Similarly, we have set $p_{\mathcal{A}} \in \{0.5, 1\}$ as the master's initial probability of auditing. The minimum probability of cheating is set to be $p_{\mathcal{A}}^{min} = 0.01$ and tolerance $\tau = 0.5$, which means that the master will not tolerate a majority of cheaters. Besides this intuition on the value of tolerance, we also carried out a set of experiments to understand the effect of this parameter, on which we will comment below.

The payoffs for the workers are set using $WB_{\mathcal{Y}} \in \{1, 2\}$ as our normalizing parameter and we take $WP_{\mathcal{C}} \in \{0, 1, 2\}$ and $WC_{\mathcal{T}} = 0.1$ as realistic values (within the same order of magnitude as $WB_{\mathcal{Y}}$) to explore the effects of these choices. The payoffs for the master are set to $MC_{\mathcal{A}} = 20$, a large value allowing us to notice the impact of auditing, and we take $MC_{\mathcal{Y}} = WB_{\mathcal{Y}} = 1$. We have set $MB_{\mathcal{R}} = 0$ and $MP_{\mathcal{W}} = 0$ to have a clear view of the master's cost. In the simulations, unless otherwise stated, the master covers all workers and not some majority as assumed in the analysis (as a worst case scenario with respect to the master's cost).

The aspiration is a parameter defined by the workers in an idiosyncratic manner; for simplicity, in these simulations we consider all workers having the same aspiration level $a_i \in \{0.01, 0.1\}$. We have checked that, when values are assigned randomly around some mean, the results are similar to those presented here, provided the variance is not very large. As for the values for the aspiration and of the workers' cost for computing the task $WC_{\mathcal{T}}$, they are such that the necessary conditions of Theorem 6 are satisfied and hence eventual convergence is reached. Finally, we consider the same learning rate for the master and the workers, i.e., $\alpha = \alpha_m = \alpha_w$. For practical reasons [48] it must be set to a small constant value, so we consider $\alpha \in \{0.1, 0.01\}$.

Convergence time: Figure 1 shows that convergence can be reached very quickly, even without punishing cheaters and with small $WB_{\mathcal{Y}}$. Note that even if all workers have the same aspiration level and begin with the same initial cheating probability, their evolution in time may be different from

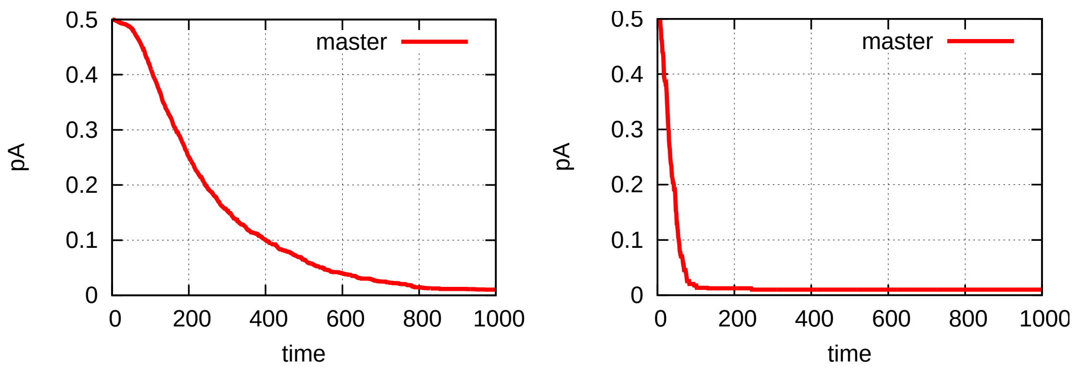

Figure 2. Auditing probability for the master as a function of time (number of rounds) for parameters $p_C = p_{\mathcal{A}} = 0.5$, $WB_{\mathcal{Y}} = 1$, $WP_{\mathcal{C}} = 0$, $WC_{\mathcal{T}} = 0.1$ and $a_i = 0.1$. Left: $\alpha = 0.01$; Right: $\alpha = 0.1$.

each other as it depends on the individual realizations of cheating. In Figure 1 we also notice that a slightly higher value of α can make the convergence time shorter. (As we argued before, the value of α can not be very high because the learning procedure will become unstable and p_C will bounce up and down without reaching convergence.) Similar conclusions can be drawn from Figure 2, where we can notice how quickly $p_{\mathcal{A}}$ drops to $p_{\mathcal{A}} = 0.01$, and also that $p_{\mathcal{A}}$ decreases in the same manner as p_C. Notice however, that $p_{\mathcal{A}}$ decreases at a slower rate; intuitively, this is to ensure that workers will not try to deviate from the desirable behavior.

Effects of punishment: Notice that in previous simulations only a positive reinforcement is applied to the workers (i.e., $WP_{\mathcal{C}} = 0$). Now, from Figure 3 we can notice that the larger the punishment we apply (i.e., $WP_{\mathcal{C}} \in \{1, 2\}$) the faster the convergence time is. In fact, we may conclude that applying only punishment is enough to have fast convergence. Comparing Figure 1(b) with Figure 3(a) we observe that, for a specific set of parameter values, a larger $WB_{\mathcal{Y}}$ leads to a shorter convergence time. Interestingly, this observation reveals a trade-off between convergence time and the cost the master has for reaching faster convergence and maintaining it. Thus, the master could choose between different protocols estimating the cost of the auditing until it reaches convergence. But less auditing leads to larger convergence times. So it is not clear initially what is going to be optimal.

Master's cost: Recall from the simulation parameters that only the cost of auditing and the workers' payment are non-zero, and that the master covers all workers (instead of some majority as in the analysis). We contrasted first a worst case scenario where workers initially cheat with probability $p_C = 1$, against the case where, given the lack of knowledge about worker's behavior, we assume that initially $p_C = p_{\mathcal{A}} = 0.5$ (refer to Figure 2).

The first conclusion we can draw from our simulations is that, even in this unfavorable situation, eventual convergence is still achieved. However, during the process the master's auditing probability reaches 1 for the system to converge. Of course, this has a direct impact on the cost of convergence, but also on the convergence time. Denote by $p_{\mathcal{A}}(0)$ the master's initial auditing probability. Interestingly, from Figures 4(a2), (a3), (b2), and (b3) we observe that, when $p_{\mathcal{A}}(0) = 1$, the convergence time decreases by 9%, and that even $p_{\mathcal{A}}$ converges to its minimum in 25% less time yielding also a cost reduction. In fact, during the first 10 rounds of evolution, for $p_{\mathcal{A}}(0) = 0.5$ the aggregate cost for the master is 56% smaller (Figure 4(a1)) than when $p_{\mathcal{A}}(0) = 1$ (Figure 4(b1)). However, in the subsequent interval between $p_{\mathcal{A}}^{min} < p_{\mathcal{A}} < 1$ the situation is reversed, and the master's cost for $p_{\mathcal{A}}(0) = 1$ is 19% smaller than the case $p_{\mathcal{A}}(0) = 0.5$. These results show that using $p_{\mathcal{A}}(0) = 1$ does not necessarily increase cost as the intuition might suggest. After convergence is achieved, Figures 4(a1), (b1), and (c1) show that, once the master's auditing probability has reached its minimum value, the master audits roughly once every hundred rounds, which is expected given that $p_{\mathcal{A}}^{min} = 0.01$. This behavior is observed independently of the initial auditing probability, which is also expected.

Another surprising result, arising from Figures 4(b3) and (c3), is that having workers with larger aspiration values makes the convergence time decrease by more than a half. The reason being that

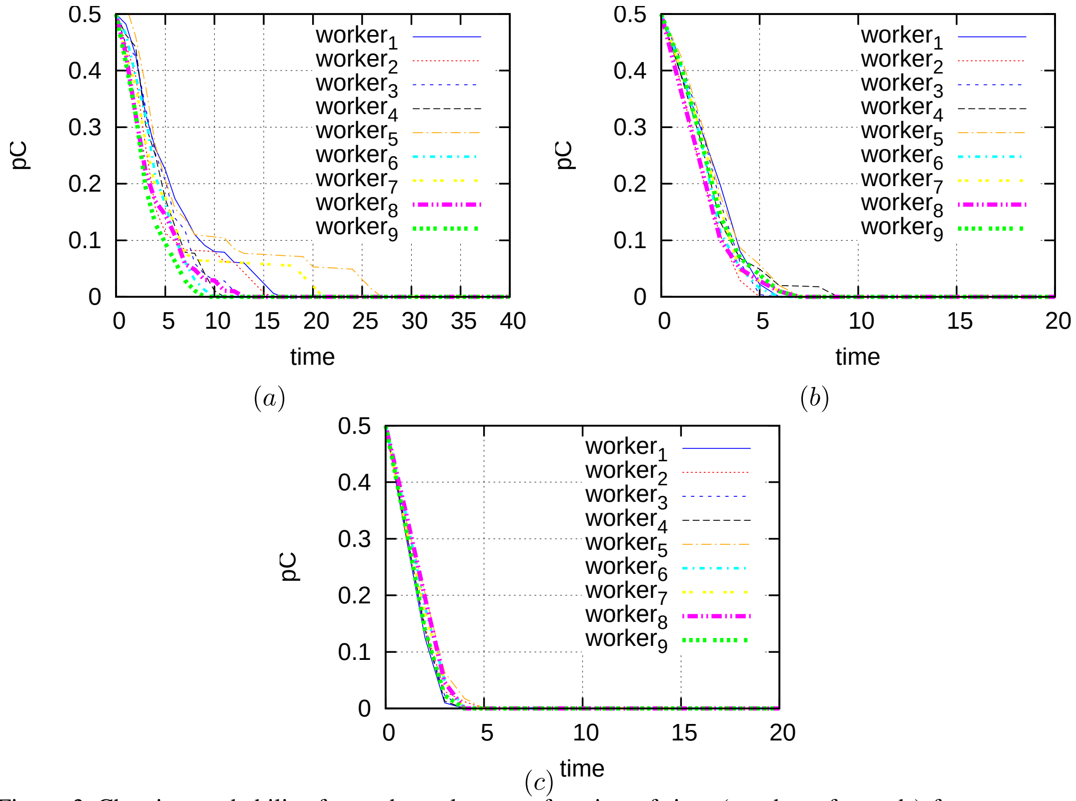

Figure 3. Cheating probability for each worker as a function of time (number of rounds) for parameters $p_C = p_{\mathcal{A}} = 0.5$, $\alpha = 0.1$, $a_i = 0.1$, $WB_{\mathcal{Y}} = 2$ and $WC_{\mathcal{T}} = 0.1$. (a) $WP_{\mathcal{C}} = 0$; (b) $WP_{\mathcal{C}} = 1$; (c) $WP_{\mathcal{C}} = 2$.

the master initially audits with probability one and all workers cheat, so a larger aspiration causes the workers' cheating probability to drop at a higher rate. This, in turn, feeds back to the master's auditing probability and cost, making them decrease faster than the case where workers have a smaller aspiration.

We have also examined the case where only a majority of workers is covered. Specifically, we have run analogous simulations to the ones depicted in Figure 4, but now covering only 5 out of the 9 workers. A first, interesting observation, is that the convergence time for the covered workers is not affected. An even more interesting observation is that the master's cost, until $p_{\mathcal{A}}^{min}$ was reached, is greater than the case of all-covered workers. This is due to the slower rate at which $p_{\mathcal{A}}$ reaches its minimum value. Of course, after this point, the master's cost is smaller since it rewards fewer workers.

Tolerance value: Finally, we have also considered the effect of tolerance for achieving eventual convergence using three different initial $p_C \in \{0.3, 0.5, 1\}$. Choosing 5000 iterations as a value large enough to illustrate the problems of convergence for high tolerance (values of the same order of magnitude or larger give basically the same results), we have found that convergence will always be reached for the lower values of p_C. This is due to the fact that with "almost" honest workers, a majority of them will always compute the answer and will force punishment to the minority of cheaters. However, when the initial p_C equals one and there is no punishment ($WP_{\mathcal{C}} = 0$), the master must respond to a percentage of cheating workers (due to tolerance) to obtain eventual convergence, and as a consequence convergence is not achieved for large tolerance values (for this specific set of parameters, when $\tau > 0.9$). Interestingly, with non-zero punishment ($WP_{\mathcal{C}} = 1$), the master can tolerate all workers cheating and still achieve convergence. Therefore, our simulated examples establish that, based on the rest of the parameter values, the master can appropriately set values for tolerance τ, (its own) learning rate α, punishment $WP_{\mathcal{C}}$ and perhaps $WB_{\mathcal{Y}}$ in such a way that convergence can be swiftly obtained.

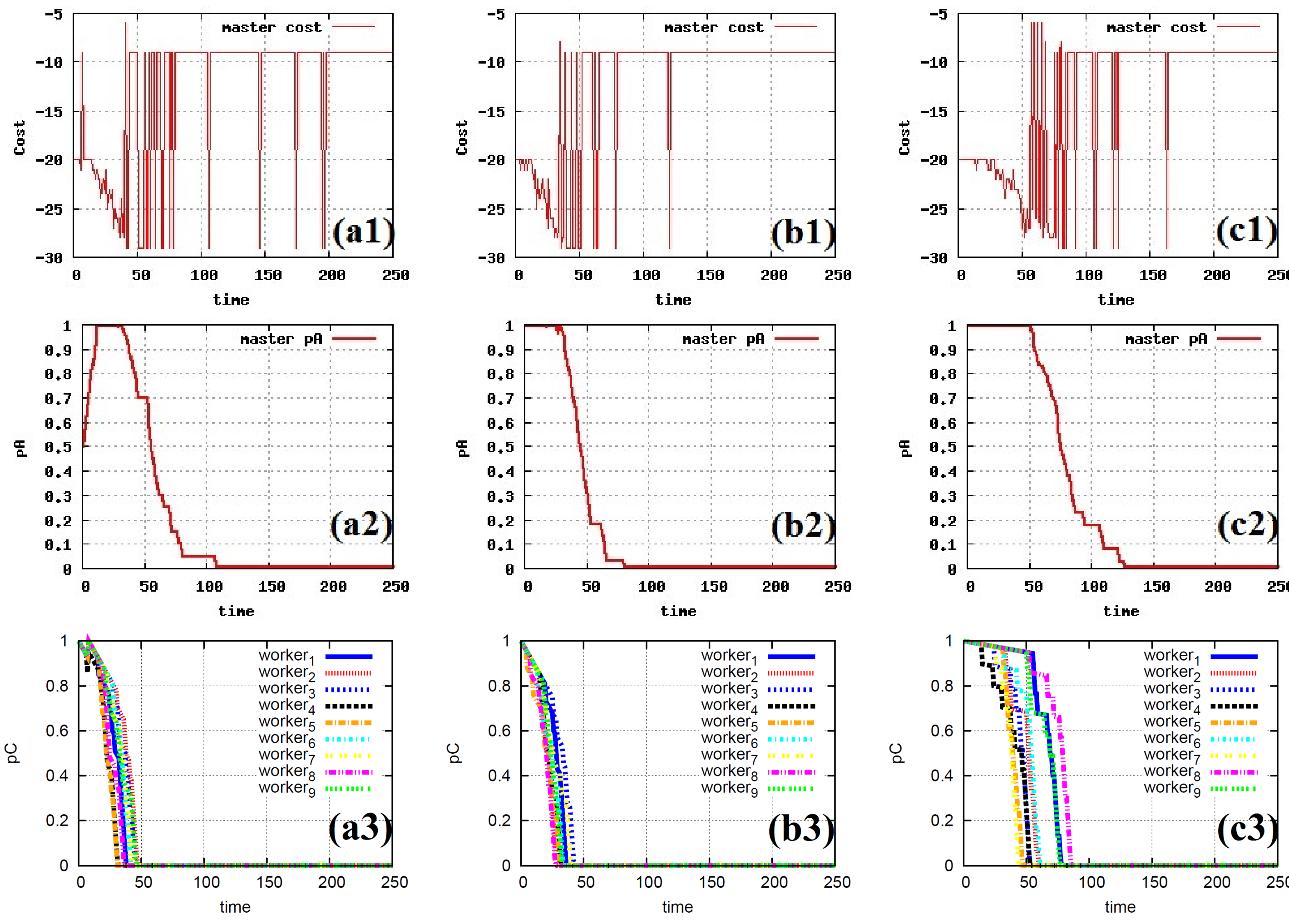

Figure 4. Top panel, master's cost as a function of time. Middle panel, master's auditing probability as a function of time. Bottom panel, worker's cheating probability as a function of time. Parameters in all panels, $p_C = 1$, $WC_{\mathcal{T}} = 0.1$, $WP_{\mathcal{C}} = 0$ and $\alpha = 0.1$. Left panel, $p_{\mathcal{A}} = 0.5$, $a_i = 0.1$. Middle panel $p_{\mathcal{A}} = 1$, $a_i = 0.1$. Right panel $p_{\mathcal{A}} = 1$, $a_i = 0.01$.

6. CONCLUSIONS AND FUTURE WORK

This work applies reinforcement learning techniques to capture the evolution of Internet-based master-worker computations. We show that under necessary and sufficient conditions, the master reaches a state after which the correct task result is obtained at each round, with minimal cost. In addition, we show that such state can be reached ''quickly''. The convergence analysis is complemented with simulations. Our simulation results suggest that when having a positive reinforcement learning (i.e., $WP_{\mathcal{C}} = 0$) the master can reach fast convergence, while applying punishments (i.e., $WP_{\mathcal{C}} \in \{1, 2\}$) provides even faster convergence. In fact, we may conclude that applying only punishment is enough to have fast convergence. Also, our simulations demonstrated that it is plausible, even if the master begins with an aggressive auditing strategy, to have a smaller cost and convergence time compared to the case of using a less aggressive auditing policy.

The analysis has provided us with provable guarantees of our approach and the simulations give an insightful view that is hard to detect from the analysis. In view of the potential of our approach, the next logical step is to do real-world experiments which is the context of our future work. However, such kind of experiment is non-trivial. We are currently in the stage of designing such real-world experiments. We expect to obtain interesting results when comparing our current approach with the observed behavior of real workers. In a follow up work, we additionally consider the presence of malicious workers that always provide the master with an incorrect answer. This will provide additional fault tolerance and security to the system.

Acknowledgments. We thank Carlos Diuk for useful discussions on reinforcement learning.

REFERENCES

1. I. Abraham, L. Alvisi, and J.Y. Halpern. Distributed computing meets game theory: Combining insights from two fields. *ACM SIGACT News: Distributed Computing Column*, 42(2):69–76, 2011.
2. 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.
3. BOINC Security, `http://boinc.berkeley.edu/wiki/BOINC_Security`.
4. Amazon's Mechanical Turk, `https://www.mturk.com`.
5. D. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer. SETI@ home: An experiment in public-resource computing. *Communications of the ACM*, 45:56–61, 2002.
6. D. Anderson. BOINC: A system for public-resource computing and storage. In *proc. of GRID 2004*, pp. 4–10.
7. M. Babaioff, M. Feldman, and N. Nisan. Combinatorial agency. In *proc. of ACM EC 2006*, pp. 18–28.
8. M. Babaioff, M. Feldman, and N. Nisan. Free riding and free labor in combinatorial agency. In *proc. of SAGT 2009*.
9. M. Babaioff, M. Feldman, and N. Nisan. Mixed strategies in combinatorial agency. In *proc. of WINE 2006*, pp. 353–364.
10. J. Bendor, D. Mookherjee and D. Ray. Reinforcement learning in repeated interaction games. Advances in Theoretical Economics, vol 1. (1), 2001.
11. J. Bendor, D. Mookherjee and D. Ray. Aspiration-based reinforcement learning in repeated interaction games: An overview. *International Game Theory Review*, 3(2-3):159–174, 2001.
12. J. Bjornerstedt and J. Weibull. Nash equilibrium and evolution by imitation. *The Rational Foundations of Economic Behavior*, 155–171, 1996.
13. T. Borgers and R. Sarin. Learning through reinforcement and replicator dynamics. *Journal of Economic Theory*, 77(1):1–14, 1997.
14. R. R. Bush and F. Mosteller. *Stochastic Models for Learning*, Wiley, 1955.
15. C. F. Camerer. Behavioral game theory: Experiments in strategic interaction. *Roundtable Series in Behavioral Economics*, 2003.
16. V. Conitzer,T Sandholm. Incremental mechanism design. In *Proc. IJCAI 2007*.
17. G. Christodoulou and E. Koutsoupias. Mechanism design for scheduling. *Bulletin of the EATCS*, 97:39–59, 2009.
18. F. G. Cross. *A theory of Adaptive Economic Behavior*, Cambridge University Press, 1983.
19. D.Easley and J. Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*, Cambridge University Press, 2010.
20. R. Eidenbenz and S. Schmid. Combinatorial agency with audits. In *proc. of GameNets 2009*.
21. "Einstein@home", `http://einstein.phys.uwm.edu`.
22. T. Estrada, M. Taufer, and D. P. Anderson. Performance prediction and analysis of BOINC projects: An empirical study with EmBOINC. *Journal of Grid Computing,* 7(4):537–554, 2009.
23. 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.
24. A. Fernández Anta, Ch. Georgiou, and M. A. Mosteiro. Designing mechanisms for reliable Internet-based computing. In *proc. of NCA 2008*, pp. 315–324.

25. A. Fernández Anta, Ch. Georgiou, and M. A. Mosteiro. Algorithmic Mechanisms for Internet-based Master-Worker Computing with Untrusted and Selfish Workers. In *proc. of IPDPS 2010*, pp. 1–11.
26. D. Fudenberg and D. K. Levine. *The Theory of Learning in Games*, MIT Press, 1999.
27. M. C. Gintis. *Game Theory Evolving*, Princeton University Press, 2000.
28. P. Golle and I. Mironov. Uncheatable distributed computations. In *proc. of CT-RSA 2001*, pp. 425–440.
29. J.Y. Halpern. Computer science and game theory: A brief survey. *Palgrave Dictionary of Economics*, 2007.
30. E.M. Heien, D.P. Anderson, and K. Hagihara. Computing low latency batches with unreliable workers in volunteer computing environments. *Journal of Grid Computing*, 7:501–518, 2009.
31. J. Henrich and R. McElreath. The evolution of cultural evolution. *Evolutionary Anthropology: Issues, News, and Reviews 12.3, 2003* ,pp. 123–135.
32. L. R. Izquierdo and S. S. Izquierdo. Dynamics of the Bush-Mosteller learning algorithm in 2x2 games. *Reinforcement Learning: Theory and Applications*, 2008.
33. K.M. Konwar, S. Rajasekaran, and A.A. Shvartsman. Robust network supercomputing with malicious processes. In *proc. of DISC 2006*, pp. 474–488.
34. 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.
35. J. Laslier, R. Topol and B. Walliser. A behavioral learning process in games. *Games and Economic Behavior*, 37:340–366, 2001.
36. A. Mass-Colell, M. Whinton, and J. Green. *Microeconomic Theory*, Oxford University Press, 1995.
37. J. Maynard-Smith. *Evolution and the Theory of Games*, Cambridge University Press, 1982.
38. J. Maynard-Smith and G. R. Price. The logic of animal conflict. *Nature*, 246:15–18, 1973.
39. J.F. Nash. Equilibrium points in n-person games. *National Academy of Sciences*, 36(1):48–49, 1950.
40. N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.
41. N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007.
42. M. A. Nowak. *Evolutionary Dynamics*. Harvard University Press, 2006.
43. S. Phelps, P. McBurney and S. Parsons. Evolutionary mechanism design: A review. *Journal of Autonomous Agents and Multi-Agent Systems*, 2010.
44. D. Rose, T. R. Willemain. The principal-agent problem with evolutionary learning. *Computational and Mathematical Organization Theory*, 2:139–162, 1996.
45. L. Samuelson. *Evolutionary Games and Equilibrium Selection*, MIT Press, 1998.
46. L. Sarmenta. Sabotage-tolerance mechanisms for volunteer computing systems. *Future Generation Computer Systems*, 18(4):561–572, 2002.
47. J. Shneidman and D.C. Parkes. Rationality and self-interest in P2P networks. In *Proc. of IPTPS 2003*, pp. 139–148.
48. C. Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool publishers, 2010.
49. M. Taufer, D. Anderson, P. Cicotti, and C. L. Brooks. Homogeneous redundancy: a technique to ensure integrity of molecular simulation results using public computing. In *proc. of IPDPS 2005*.
50. J.W. Weibull. *Evolutionary Game Theory*, MIT Press, 1995.
51. “WUProp@home”, `http://wuprop.boinc-af.org/`.
52. 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.