X

Optimal Contracts for Outsourced Computation

Viet Pham, MHR. Khouzani, Carlos Cid, Royal Holloway University of London
{viet.pham.2010,arman.khouzani,carlos.cid}@rhul.ac.uk

While expensive cryptographically verifiable computation aims at defeating malicious agents, many civil purposes of outsourced computation tolerate a weaker notion of security, i.e., "lazy-but-honest" contractors. Targeting this type of agents, we develop optimal contracts for outsourcing of computational tasks via appropriate use of rewards, punishments, auditing rate, and "redundancy". Our contracts provably minimize the expense of the outsourcer (principal) while guaranteeing correct computation. Furthermore, we incorporate practical restrictions of the maximum enforceable fine, limited and/or costly auditing, and bounded budget of the outsourcer. By examining the optimal contracts, we provide insights on how resources should be utilized when auditing capacity and enforceability are limited. Finally, we present a light-weight cryptographic implementation of the contracts and discuss a comparison across different implementations of auditing in outsourced computation.

Key Words and Phrases: Outsourced Computing, Moral Hazard, Optimization, Game Theory

ACM Reference Format:
Viet Pham, MHR Khouzani and Carlos Cid, 2014. Optimal contracts designed for outsourcing of computational tasks via appropriate use of rewards, punishments, auditing rate, and redundancy scheme. *ACM* X, X, Article X (February 2014), 27 pages.

1. INTRODUCTION

The idea of outsourcing complex computation tasks has been proposed and implemented in a variety of applications. Research projects involving complex analysis on a huge multitude of data have utilized parallel processing of their computations on the processors of millions of volunteering Internet users. These include search for extra-terrestrial life (*SETI@Home*), investigation of protein folding and computational drug design (*Folding@Home* and *Rosetta@home*). Businesses from different sections including finance, energy infrastructure, mining and commodities transport, technology and innovation [Fullbright 2011] have also realized the benefits of outsourcing their data and computation, and "moving to the cloud". The cloud, as a dedicated infrastructure with specialized man-force and powerful computing capabilities, along with the ability to pool demands from different clients and dynamic assignment of the resources can reduce the cost of computation. On the other hand, the outsourcer is also relieved from the dedicated investment in its computing infrastructure and in addition, has the total flexibility of pay-per-use paradigm, to flex on or to flex-off services effortlessly [Fullbright 2011]. The growing trend of outsourced computing have made possible small virtualized computers and smart devices with powerful computational power, with applications in critical mission scenarios as well as everyday use.

In all of these scenarios, there is a concern for the outsourcer (client) about the correctness of the returned results. The provider of computation services (the servers) have an economic incentive to return guessed results as opposed to performing the computation completely and honestly, and thereby save on the computation work. Hence, to make this paradigm viable and guarantee soundness of the results, there must be an auditing mechanism in place. The auditing, however, is not free: it either creates computational overhead for the client, the server, or both. The auditing can be done by the outsourcer itself or through a trusted third party for a fee, say, through re-computation. Alternatively, a *redundancy* scheme can be employed in which the same job is outsourced to multiple servers and the results are checked against each other.

Irrespective of the auditing mechanism, the outsourcer can set an extremely large fine for detected wrong results, and make cheating theoretically impossible even for the lowest probability of cheat detection. However, in practice, an extremely large fine is a non-credible threat. A more reasonable assumption is a cap on the maximum enforceable fine, with the special interesting case where the cap is zero. In this paper we provide a concrete and general approach based on Principal-Agent modelling from game theory to optimal contract designs for outsourcing from the client (principal) to the servers (agents). Specifically, we assume a general maximum enforceable fine, maximum budget, and costly and/or limited auditing rate. We formulate the utilities of both the principal and the agents, as well as essential constraints that guarantee honest computation (incentive compatibility) along with their acceptance of the offer (participation). This allows us to effectively and systematically compute the optimal contract such that the principal's expense is minimized. Our work hence provides a benchmark enabling meaningful comparison among different deployments of computation outsourcing.

The paper is structured as follows: In Section 2, we briefly overview previous results in relation to our approach and describe our contributions. This is followed by a detailed motivation of our contract model in Section 3, along with descriptions of important constraints that make the problem non-trivial. In Section 4, we compute optimal contracts involving only one agent, and explore related improvements. In Section 5, we allow the principal to also potentially outsource the same task to multiple non-colluding agents as an alternative means of auditing and develop optimal hybrid contracts. We further establish the global optimality of our hybrid two-agent contracts among all possible contracts involving any number of non-colluding agents with respect to the notion of Nash Equilibria. In Section 6, we comment on cryptographic implementation of our contracts, i.e., how to enforce the terms and policies in an automated way. We also briefly discuss and contrast different techniques of auditing besides simple recomputation. Finally, in Section 7, we conclude the paper with a summary of the results and remark on some potential future directions.

2. RELATED WORKS

A line of research is focused on designing reliable verification techniques for outsourcing of special-purpose computations. For instance, [Wang et al. 2011] investigates outsourcing of linear optimizations. Other notable examples are queries on outsourced databases, including typical queries [Atallah et al. 2008; Chen et al. 2008] and aggregation [Yi et al. 2009]. Their main paradigm is for the querier to rely on trusted information directly given by the data owner (outsourcer) to verify the results returned by the servers.

Verification methods for general-purpose computing also appear in several remarkable works. In [Monrose et al. 1999] verification is performed by re-executing parts of the computation. A variation is presented in [Canetti et al. 2011] in which the authors utilize redundancy over multiple agents, assuming that at least one of them is honest. Outsourced computation has also caught attraction in cryptographic research: in a seminal work, the authors of [Gennaro et al. 2010] formally define verifiable computation and give a non-interactive solution. Their solution uses Yao's garbled circuits to represent the computation and homomorphic encryption to hide such circuits from the agents. More efficient but interactive solutions that use *probabilistically-checkable proofs* (PCPs) have since been developed such as PEPPER [Setty et al. 2012a] and GINGER [Setty et al. 2012b].

Incentive-based solutions such as [Belenkiy et al. 2008; Nix and Kantarcioglu 2012] have studied contracts that the outsourcer may offer to the agents and through a combination of auditing, fines and rewards, honest computation is enforced. All of these

verification techniques are, however, costly in terms of computation, memory, incentive rewards, etc., either to the prover or the verifier, or both. For example, the scheme in [Monrose et al. 1999] requires partial re-execution of the tasks, and the verification in [Canetti et al. 2011] incurs cost in the redundancy of the number of computing agents. Also, efficient protocols like PEPPER still incurs a cost in the order of m^3 [Setty et al. 2012a] on the principal, where m is the size of the problem. The cost of employing verifiable computing across these different schemes hence raises the important question of how to use them economically, especially when there is a flexibility in parameters that govern the overall cost to the outsourcer. Motivated by this question, we abstract the verification techniques as an auditing tool with a exogenous cost and provide incentive-based contracts that minimize the expected cost of the principal. Our model can be applied to any special-purpose or generic verification scheme. Our contributions generalize the results in [Belenkiy et al. 2008; Nix and Kantarcioglu 2012] by (1) extending the feasibility of honesty enforcing schemes for *any* bound on the enforceable fines and *any* auditing capacity; (2) explicitly accounting for the cost of auditing and treating the auditing rate as one of the choice variables; and (3) providing optimal contract that minimize the aggregate cost of the principal as a combination of incentive payments and auditing costs. In short, our work explicitly extends both applicability and efficiency of incentive-based solutions based on a general abstraction of the verification method employed.

3. PROBLEM DEFINITION: GENERAL SETUP

In this section, we describe the general setting of the problem and basic assumptions behind our model. A list of the main notations is provided in Table I for reference.

The outsourcer, which we refer to as the *principal*[1] has a deterministic *computation task* to be executed to obtain the output (result). Instead of executing the task itself, the principal hires a set of *agents*[2] to do this. The principal aims to enforce *fully honest* computation of the task through setting a contract, involving rewards, auditing, and punishments (fines).

The principal and the agents are each selfish non-cooperative expected utility maximizers. Initially, we assume that everybody is risk-neutral, i.e., they have no strict preference between their expected utility and their utility of expected reward, and hence [Gintis 2009, ch.2.4], their utilities are linear function of the costs (with negative sign) and the rewards (with positive sign). Moreover, we assume that agents are "lazy but not malicious", that is, they do not have any interest in potentially reporting dishonest computations other than saving in their computation cost. Suppose the range and the probability distribution of the computation result is known. Generating a guessed output according to this distribution has zero computation cost and accuracy probability of q_0 (which can be negligibly small if the range of the output is large). For the sake of generality, as in [Belenkiy et al. 2008], suppose each agent also has access to a private and independent *tricky algorithm* Alg that generates the correct output with probability q_1, where $q_0 < q_1 < 1$, at the cost of $c(q_1) \geq c(q_0) = 0$. The cost of honest computation is $c(1)$, which is strictly greater than $c(q_1)$. To enforce honesty of the agents, the principal *audits* the returned result with probability λ. We assume that auditing is perfect, i.e., if the output is indeed correct, the audit definitely confirms it (no "false positives"), and if the output is incorrect, the audit surely detects it (no "false negatives"). In the most basic contract, the principal decides on an auditing rate λ, sets

[1]Also called the *boss* [Belenkiy et al. 2008], *master* [Christoforou et al. 2013], *outsourcer* [Carbunar and Tripunitara 2012], *client* [Gennaro et al. 2010], *data owner* [Nix and Kantarcioglu 2012], etc.
[2]Also referred to as the *workers*, *servers*, *clouds*, or *contractors*.

a penalty (fine) f for detected erroneous answers and reward r otherwise. What makes the problem non-trivial is the following list of observations:

(1) *Costly detectability of cheating*: that auditing *all* of the results is either *infeasible* or *undesirable*. Regarding the infeasibility, suppose that in the long run the principal has a continuous demand (e.g. the Folding@Home project) of tasks awaiting computation, appearing at a rate ρ tasks per unit time. Also, suppose that each audit takes the principal ν machine cycles, and the computation capacity of the principal's machine is κ cycles per unit time. Then the maximum feasible rate of verification is $\frac{\kappa}{\nu\rho}$.[3] Moreover, auditing (e.g. through re-computation) may be costly as it will consume the computation power of the principal's machine and slow it down, or it will require obtaining additional hardware. The principal chooses the probability of auditing of a task $\lambda \in [0, \Lambda]$, where $0 < \Lambda \leq 1$ is associated with the computational capacity of the principal. The principal incurs the cost $\Gamma(\lambda)$ which is non-decreasing in λ. For simplicity of exposition, we assume a linear relation: $\Gamma(\lambda) = \gamma\lambda$ for a given $\gamma \geq 0$. An alternative to the occasional redoing of the whole computation by the principal can be using a third-party cloud that is highly reliable but costly (with per access cost of γ). For this scenario, the maximum auditing rate Λ is one, i.e., all of the tasks could be audited, albeit at an excessive cost.

(2) *Limited enforceability of the fines*: The problem of verifiable computing could become trivial if there is no bound on the fine that can be practically levied on a wrongdoer: as long as there is even a tiniest probability of detection, then the principal can make the expected utility of the smallest likelihood of cheating become negative by setting the fine for erroneous results large enough. The issue with this argument is that such a fine may be extremely large and hence, become an *incredible threat*, in that, if the cheating of an agent is indeed caught, the fine is practically or legally non-collectable. Thus, existence (feasibility) results of honesty enforcement that rely on choosing a "large enough" fine are rather straightforward and uninteresting. In particular, such approaches leave unanswered the question of whether honest computation is still attainable for a bounded enforceable fine below their prescriptive threshold. Moreover, such results do not provide a good metric of comparison between alternative incentive schemes, or across different choices of parameters for a particular scheme. We will explicitly introduce $F \geq 0$ in our model to represent the maximum enforceable fine and obtain the optimal contracts subject to $f \leq F$. This can be the "security deposit", prepaid by the agent to the principal, that is collectible upon a provable detection of an erroneous result. A special case of interest is $F = 0$, i.e., when the only means of punishment is refusal to pay the reward.

(3) *Limited budget*: As with the maximum enforceable fine to make it a credible threat, the maximum instantaneous "budget" of the principal leads to a bound on the reward to make it a credible promise. Let the maximum instantaneous payable reward by the principal be R. Thus, we require: $r \leq R$.

4. CONTRACTS FOR SINGLE AGENT

In this section, we consider the case where the contract is designed for and proposed to only one computing agent. We provide the optimal contract for the basic model in

[3]Note that even when the principal is verifying at full capacity, it should not pick the next immediate task to verify after finishing the previous one, since it may create a "learnable" pattern of audited tasks, which the agent can use to only be honest when computing them. This however can be avoided if the principal picks uniformly randomly tasks at the rate of $\frac{\kappa}{\nu\rho}$ and store them in a queue. However, the practical buffer has a storage limit. Consequently, the maximum feasible auditing rate with no essential pattern is strictly less than the full capacity rate $\frac{\kappa}{\nu\rho}$.

Table I: List of main notations

parameter	definition
λ	probability of auditing an outsourced computation by the principal
Λ	the physical upper-bound on λ
γ	cost of auditing (incurred by the principal)
q	probability of a correct computation by the agent
q_0	the correctness probability of a random guess from the output space
$c(q)$	the expected cost of computation to an agent for the correctness level of q
$c(1), c$	cost of an honest computation to an agent
f	fine collected from agent upon detection of an erroneous computation
F	the maximum enforceable fine
r	reward to the agent for an unaudited or audited and correct computation
R	the maximum feasible reward
z	the reserve utility (a.k.a., fallback utility or aspiration) of the agent
H	auxiliary coefficient defined as $c(1) + z$ (§4)
K	auxiliary coefficient defined as $(c(1) - c(q_1))/(1 - q_1)$ (§4)
$\mathcal{C}$	the expected cost of the contract to the principal
α	probability of using two agents for the same computation (§5.1)
F_0	auxiliary coefficient defined as $c/\Lambda - c$ (Proposition 5.1, §5.1)
F_1	auxiliary coefficient defined as $c[c-\gamma]^+/[2\gamma - c]^+$ (Proposition 5.1, §5.1)

subsection 4.1. In subsection 4.2, we investigate what happens if the risk-neutrality assumption of the agents is relaxed. Next in subsection 4.3, we comment on moderating against using tricky algorithms and clever guesses. Subsequently, in subsection 4.4, we discuss the optimal choice of the principal in the light of the optimal contracts theretofore developed. We close the case of single-agent in subsection 4.5 by generalising our results to contracts in which the principal is allowed to reward unaudited and verified tasks potentially differently. In Section 5, we will investigate the multi-agent case.

The action of the agent, given the parameters of the contract set by the principal, is first whether to accept it, and if so, which (probabilistic) algorithm to choose for computation of the assigned task. Since a naive random guess is correct with probability q_0, we assume that the agent's algorithm is correct with probability $q \in [q_0, 1]$. Let u_A denote the expected utility of the agent after accepting the contract. With correctness probability of q, the agent is caught (and fined) with probability $(1-q)\lambda$. Hence, u_A is composed of expected reward $[1-(1-q)\lambda]r$, minus the expected cost composed of the cost $c(q)$ of the agent's algorithm and the expected fines $(1-q)\lambda f$. Hence: $u_A(q) = [1-(1-q)\lambda]r - c(q) - (1-q)\lambda f$. The agent may be able to achieve the same level of correcness, i.e., q, with different randomizations between the tricky algorithm Alg, the honest algorithm and random (naive) guessing. It is sraigtforward to make the following observation: For any q, the best $c(q)$ is achieved as follows:

a) If $[c(1) - c(q_1)]/(1-q_1) > c(1)/(1-q_0)$, then: $c(q) = \begin{cases} L_{q_0,q_1}(q) & q_0 \le q \le q_1 \\ L_{q_1,1}(q) & q_1 \le q \le 1 \end{cases}$;

b) If $[c(1) - c(q_1)]/(1-q_1) < c(1)/(1-q_0)$, then: $c(q) = L_{q_0,1}(q)$, where in both cases, $L_{x,y}(z) := c(x) + \dfrac{c(y)-c(x)}{y-x}(z-x)$, i.e., the linear combination of the costs of the corresponding two end points.

Note that in case-(b), the risk-neutral agent would never use Alg, since the cost of using it can be undercut (in expected value) by randomizing between honest computation and random guessing. Hence, we only consider case-(a) for now and revisit case-(b) in §4.3.

4.1. Optimum Contract for a Single Agent

The principal chooses the contract by setting the rate of auditing and reward and punishment values, in order to maximize its own utility and ensure *fully honest* computation. Hence, the reward and punishments, r and f, should be chosen such that honest computation is the optimal course of action for the agent, if the contract is accepted. This means ensuring: $1 = \arg\max u_A(q)$. Following the Principal-Agent modelling in game theory (e.g. [Gintis 2009, ch.7] or [Rasmusen 1994, ch.6]), we will refer to this as the *incentive compatibility* constraint. For case (a), this becomes:

$$u_A(1) = r - c(1) \geq u_A(q_1) = [1 - (1 - q_1)\lambda]r - c(q_1) - (1 - q_1)\lambda f \tag{1}$$

The agent accepts the contract if its expected utility is larger than its *reserve utility*, $z \geq 0$.[4] Given incentive compatibility, this *participation constraint* is hence:[5]

$$u_A(1) = r - c(1) \geq z. \tag{2}$$

The principal wants to get away with the least reward and auditing rate. Therefore, the *optimal* contract for the single agent reduces to solution of the following optimization:

$$\min_{r,f,\lambda} \mathcal{C} := r + \gamma\lambda \tag{3a}$$

$$s.t. \quad r \leq R,\ 0 \leq f \leq F,\ 0 \leq \lambda \leq \Lambda, \tag{3b}$$

$$r \geq H,\ r\lambda + f\lambda \geq K \tag{3c}$$

where (3c) is derived from (1) and (2) in which we have used the auxiliary coefficients $H := c(1) + z$ and $K := [c(1) - c(q_1)]/(1 - q_1)$ for brevity. Then (proof in Appendix-A):

PROPOSITION 4.1. *With the parameters given in Table I, the contract that enforces honest computation and is accepted by the agent, and minimizes the cost of the principal is by setting $f^* = F$ and choosing λ^*, r^* as given by the following:*[6]

$$\gamma \leq \frac{K}{\Lambda^2}: \begin{cases} [\frac{K}{\Lambda} - H]^+ \leq F: & \lambda^* = \frac{K}{H+F},\ r^* = H,\ \mathcal{C}^* = H + \frac{\gamma K}{H+F} \\ [\frac{K}{\Lambda} - R]^+ \leq F < [\frac{K}{\Lambda} - H]^+: & \lambda^* = \Lambda,\ r^* = \frac{K}{\Lambda} - F,\ \mathcal{C}^* = \frac{K}{\Lambda} + \gamma\Lambda - F \end{cases}$$

$$\gamma > \frac{K}{\Lambda^2}: \begin{cases} [\sqrt{K\gamma} - H]^+ \leq F: & \lambda^* = \frac{K}{H+F},\ r^* = H,\ \mathcal{C}^* = H + \frac{\gamma K}{H+F} \\ [\sqrt{K\gamma} - R]^+ \leq F < [\sqrt{K\gamma} - H]^+: & \lambda^* = \sqrt{\frac{K}{\gamma}}, r^* = \sqrt{K\gamma} - F, \mathcal{C}^* = 2\sqrt{K\gamma} - F \\ [\frac{K}{\Lambda} - R]^+ \leq F < [\sqrt{K\gamma} - R]^+: & \lambda^* = \frac{K}{R+F},\ r^* = R,\ \mathcal{C}^* = R + \frac{\gamma K}{R+F} \end{cases}$$

For $F < [\frac{K}{\Lambda} - R]^+$, the optimization is infeasible, i.e., there is no honesty-enforcing contract that is also accepted by the agent.

Discussion. The first observation is that the optimal contract should fully utilize the maximum enforceable fine and punish at no less than F. For large values of enforceable fines, we note that r^* is at H, the minimum value to ensure participation, and

[4]The reserve utility (also referred to as the *fall-back utility* or *aspiration wage*) is the minimum utility that the agent aspires to attain or can obtain from other offers. Naturally, $z \geq 0$. Note that an implicit assumption here is that the agent is replaceable by any other agent with the same fall-back utility, i.e., there are many agents available with the same reserve utility. Without this assumption, the agent has negotiation power by refusing the contract knowing that it cannot be replaced. Alternatively, z can be thought as to (exogenously) capture the negotiation power of the agents. This is an assumption we make throughout the paper.

[5]Participation constraint is sometimes also called Individual Rationality constraint.

[6]The notation $x^+ := \max\{0, x\}$.

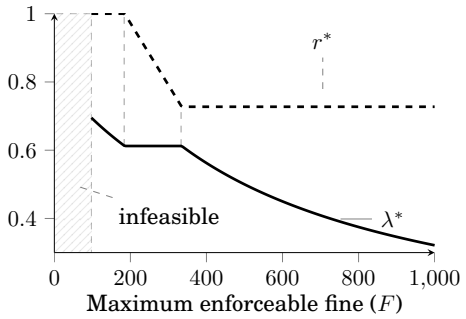

Fig. 1: Change of contract parameters r^*, λ^* w.r.t. the maximum enforceable fine F (Prop. 4.1, case of $\gamma > \frac{K}{\Lambda^2}$), where $K = 450$, $\gamma = 1200$, $\Lambda = 0.7$, and $c = 400$.

$\lim_{F\to\infty} \lambda^* = 0$, which yields $\lim_{F\to\infty} \mathcal{C}^* = H$. These are compatible with intuition as a huge fine implies that honesty can be enforced with minimum compensation and minuscule rate of inspection. When auditing is cheap ($\gamma \leq K/\Lambda^2$), increasing the auditing rate is the better option to compensate for lower values of F to maintain incentive compatibility (honest computation). This is unless the auditing rate is at its maximum Λ, in which case, reward must increase above H to maintain incentive compatibility and compensate for the low value of F. Note that in this case, the participation constraint is not active and is satisfied with a slack, while the incentive compatibility constraint is satisfied tightly. For yet lower values of enforceable fine F, even maximum reward $r = R$ and auditing rate $\lambda = \Lambda$ might not impose a strong enough threat against cheating, hence the infeasibility region. When auditing is expensive ($\gamma > K/\Lambda^2$), in order to retain incentive compatibility in the situation of very low fine F, the principal should increase reward, and only consider more frequent auditing if the reward budget R has been reached. Fig. 1 depicts the optimal parameters of the contract versus the maximum enforceable fine for the latter case ($\gamma > K/\Lambda^2$).

Note that the infeasible region does not necessarily exist. Specifically, when the principal's instantaneous budget R is larger than K/Λ, then there is always a feasible contract. Then even for $F = 0$, i.e., no enforceable fine, a contract that enforces honest computing is feasible, albeit by using high values of reward and/or auditing rate. In such cases, the principal "punishes" audited erroneous computations only through not rewarding the agent. From the proof, however, it becomes clear that honesty cannot be enforced with zero auditing rate, and consequently, the case of $\Lambda = 0$ trivially leads to infeasibility. Moreover, to satisfy the participation constraint at all, R has to be at least as large as H. Hence, for $R < H$, likewise, there exists no feasible contract for any F. The proof also shows that except for the special case of $\gamma = 0$, the optimal contract has the feature that it is *unique*.

Figures 2a and 2b depict the change in the structure of the optimal contract versus varying auditing cost γ and the maximum auditing capacity, respectively. From Fig. 2a, we can see that for larger values of γ, the optimal contract utilizes lower values of inspection rate λ^* while using higher values of reward r to enforce honest computation. This transition progress culminates when the payment reaches its threshold R, after which the contract remains unchanged. In contrast, Fig. 2b shows how increasing the maximum auditing capacity affects the optimal contract in the opposite trend: as the principal is more capable of auditing, it should consider more frequent auditing and lessen the reward for honest computation. The payment, however, can never be lowered below H to maintain participation.

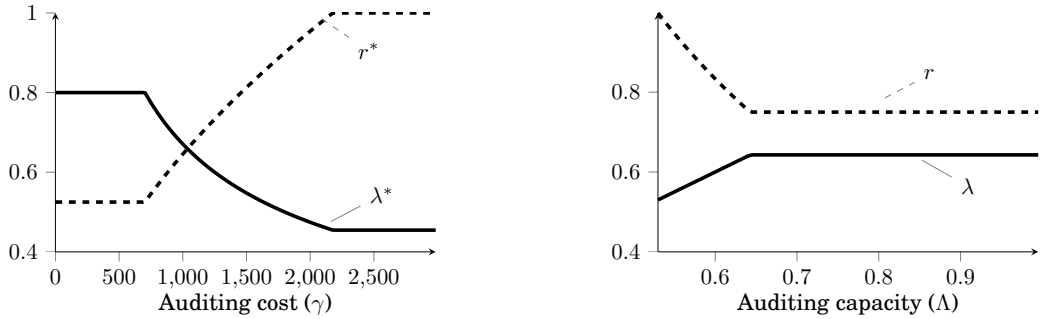

Fig. 2: Optimal contract parameters w.r.t (a) the auditing cost γ, with $K = 450$, $\Lambda = 0.8$, $c = 400$, and (b) auditing capacity Λ, with $K = 450$, $\gamma = 450$, $c = 450$.

4.2. A Risk-Averse Agent

So far, we modelled the agent as risk-neutral, i.e., one that is indifferent between its expected utility and utility of expectation, leading to a linear utility function. However, empirically, individuals tend to show risk-aversion regarding decisions that affect their income. By definition, (strict) risk aversion is (strict) preference of expected utility over utility of expectation. Following *Jensen*'s inequality, this is equivalent to assuming a (strictly) concave utility function (ref. e.g. [Gintis 2009, ch.2.4]). We have the following simple but re-assuring result:

PROPOSITION 4.2. *The optimal contract given in Proposition 4.1 developed for a risk-neutral agent stays feasible for any risk-averse agent as well.*

The proof is provided in Appendix-B. Note that even though the feasibility of our contract is guaranteed, its optimality might no longer hold. This is because a lower value of fine and/or rewards could potentially maintain incentive compatibility, as intuitively, cheating with a chance of getting caught can be seen as a lottery. However, because the level of risk-averseness of an agent is unknown, we argue that it is best practice to design the optimal contract for the worst case with respect to risk, i.e., risk neutrality. Specially, if a contract is designed assuming a particular degree of risk-aversion of the agent but the agent turns out to be less risk-averse than assumed, then the incentive-compatibility for honest computation may be violated, failing the principal's intolerance of erroneous computations. Accordingly, for the rest of the paper, we will retain risk-neutrality for agents.

4.3. Mitigating clever guesses

An inherent problem of outsourced computation is that a (not always) correctly guessed output is indistinguishable from an honestly computed one. For instance, consider the question of whether a large natural number is a prime: the deterministic guess of "no" is most likely correct. Also, since the principal might not know the exact cost and success probability of potential guessing algorithms, it is hard to design a contract that enforces honesty. Therefore, the principal may prefer to avoid identifying the parameters of guessing algorithms altogether.

One way to mitigate the possibility of "clever" guesses is to enlarge the output range by requiring the agent to return not just the final computation output, but also snapshots of intermediate steps of the computing process [Belenkiy et al. 2008]. This will reduce the correctness probability of a naive guess down to $q_0 = \mathsf{negl}$. Moreover, requir-

ing snapshots of the intermediate steps makes guessing of the correct output more costly. Let $c(q_1)$ be the cost of a tricky algorithm that tries to produce the expanded output with the intermediate steps of the honest computation, where it succeeds with probability q_1. We make the assumption that now $c(q_1) > q_1 c(1)$, so that any guessing algorithm with cost $c(q_1)$ can be replaced with a randomization between naive guess (with weight $1 - q_1$) and honest computation (with weight q_1). Thus, for incentive compatibility, we only need to make sure that the agent's utility from honest computation is better than a naive guess that succeeds with negligible probability $q_0 = \mathsf{negl}$. To avoid distraction in our analysis, we assume $q_0 = 0$, as the results can easily be realized for $q_0 = \mathsf{negl}$. Our simplified constraints for the contract become:

$$\text{participation}: \ r \geq c(1) + z, \ \text{incentive compatibility}: \ r \geq \frac{1}{\lambda} c(1) - f. \tag{4}$$

Comparing to the constraints in (3c), this translates to changing K to $c(1)$. This in turn implies that the new incentive compatibility constraint requires a strictly lower fine value. Intuitively, as guessing becomes more difficult, cheating becomes less attractive and hence can be deterred with a smaller fine. Hereafter, we assume that the principal is employing this technique and use the above incentive compatibility constraint. Moreover, for simplicity of exposition, we assume that the reserve utility z is zero, and hence H becomes $c(1)$, which we will abbreviate as c.

4.4. Optimal Choice for the Principal

So far we have considered auditing as a blackbox and only included its cost and capacity into the model. However, when auditing is via redoing the computation (at the cost of γ) it might be optimal for the principal to not offer any contract at all. Indeed, when $\Lambda = 1$, the principal can potentially audit all computations by redoing them. Specifically, if the optimal contract costs $\mathcal{C}^* \geq \gamma$, then it is optimal for the principal to do the computation itself, as that only costs $\gamma\Lambda = \gamma$. In case $\Lambda < 1$, the principal cannot *do* all the computations, and must outsource a portion of it. Interestingly, the following proposition establishes that the principal's optimal choice is either to not outsource at all, or fully outsource its computation.

PROPOSITION 4.3. *Consider the case where auditing is through redoing the computation. Let x be the probability that the principal computes the tasks itself. Then, either $x^* = 0$ and the optimal contract is as per Proposition 4.1, or $x^* = \Lambda = 1$ and there should be no outsourcing.*

The proof is in Appendix-C. The proposition has this important corollary:

COROLLARY 4.4. *When $\Lambda < 1$, the optimal choice for the principal is to use the optimal contact given by Proposition 4.1. When $\Lambda = 1$, the optimal choice of the principal is to compare the expected cost achieved by the optimal contract in Proposition 4.1 (for the value of maximum enforceable fine at hand) against γ, and accordingly decide to outsource or independently compute all of the tasks.*

4.5. Optimal Contract for a Single Agent: Two-Level Reward

In our contracts so far, verified correct results and unaudited results are rewarded identically at r. Suppose, alternatively, that the principal rewards r_0 for accepted but not audited results and r_1 for corroborated correct answers, and as before, penalizes f for detected wrong computations. This way, the principal may hope to save significantly by, for example, not paying for unaudited computations. The new incentive compatibility and participation constraints are: $(1-\lambda)r_0 + \lambda r_1 - c \geq (1-\lambda)r_0 - \lambda f$ and $(1-\lambda)r_0 + \lambda r_1 - c \geq 0$, respectively. The optimization of (3) for a contract with two-level

reward changes to:

$$\min_{r_0,r_1,f,\gamma} \mathcal{C} := r_1\lambda + r_0(1-\lambda) + \gamma\lambda$$
$$s.t. \quad r_0, r_1 \leq R,\ f \leq F,\ 0 \leq \lambda \leq \Lambda,\ r_1\lambda + r_0(1-\lambda) \geq c,\ r_1 \geq \frac{c}{\lambda} - f.$$

PROPOSITION 4.5. *For $F \geq [c/\Lambda - R]^+$, the optimal single-agent contract for two-level rewarding is given as: $f^* = F$, $\lambda^* = c/(F+R)$, $r_1^* = R$, $r_0^* = Fc/(R-c+F)$, $\mathcal{C}^* = c\,(1 + (\gamma + c - R)/(F+R))$. For $F < [c/\Lambda - R]^+$, the contract is infeasible.*

The proof is similar to that of Proposition 4.1, and is omitted for brevity.

Discussion of the two level reward contract. First, note that there is no improvement in terms of the infeasibility region compared with the single-level reward contract. However, the achieved cost is always better. This was to be expected as the single-level rewarding can be thought of as a special case of two-level. However, the behaviour of the optimal contract now does not depend on the value of the auditing cost γ. This is where the strength of the two-level rewarding lies: for high values of γ, the two-level contract increasingly outperforms the single reward-level contract.

Note that the optimal reward for audited and correct results r_1 is at the principal's maximum budget R irrespective of the value of F. The value of reward for unaudited results r_0 is always strictly less than c, i.e., the cost of honest computation (and hence strictly less than r_1 as well). The value of r_0, unlike r_1, depends on F: For higher values of maximum enforceable fine, in fact somewhat unexpectedly, the optimal contract chooses increasing values of reward r_0^*. Still intuitively, a larger threat allows less necessity for auditing, and thus the contract starts to behave as a "lottery", in which the low-chance "winner" receives $r_1^* = R$ and the "loser" $r_0 < c < R$.

5. OPTIMAL CONTRACTS FOR MULTIPLE AGENTS

When there are more than one agent available, the set of possible contracts gets extended. Specifically, as e.g. [Belenkiy et al. 2008] and [Nix and Kantarcioglu 2012] discuss, the principal has the option of submitting the same task to multiple agents and comparing the outcomes. We will refer to this option as the *redundancy* scheme. If the returned results do not match, it is clear that at least one agent is cheating. Furthermore, as [Nix and Kantarcioglu 2012] assumes, if the agents are non-colluding, and returning the intermediate steps along with the computation result is required, then the probability that the results produced by cheating will be the same will be negligible, which we again assume to be zero (for simplicity). Hence, the returned results are correct *if and only if* they are the same.

In the next subsection, we develop optimal contracts considering two agents. Subsequently, we establish the global optimality of two-agent contracts among any number of agents with respect to the notion of Nash Equilibrium.

5.1. Optimal Contracts for Two Agents

Consider the case that there are two agents available: agent 1 and 2. As in the single-agent case, consider a principal that has a computation task and a maximum auditing rate of Λ. Then, in general, a principal can use a hybrid scheme: it may choose to send the same job to both of the agents sometimes, and to one randomly selected agents the rest of the time. Sending the same task to two agents provides a definite verification, however, at the cost of paying twice the reward, since both agents must be rewarded for honest computation. Hence, an optimal choice of redundancy scheme is not immediately clear, even less so if this schemes is randomized with just choosing one agent

and doing independent audits. In this section, we investigate optimal contracts among all hybrid schemes.

Besides lack of collusion, we assume the agents do not communicate either. Therefore, on the event that any of the agents receives a task, it has no information about the busy/idle state of the other agent. The action of each agent is selection between honest computation, which we represent by $\mathscr{H}$, and cheating, which we denote by $\mathscr{C}$. Since the agents have no information about the state of the other agent, the set of their (pure) strategies and actions are the same.

The expected utility of each agent depends in part on the action of itself and of the other agent. Let $u_A(a_1, a_2)$ represent the utility of agent 1 when it chooses action a_1 and agent 2 chooses a_2, where $a_1, a_2 \in \{\mathscr{H}, \mathscr{C}\}$. The principal wants to enforce honest computation with probability one. If $u_A(\mathscr{H}, \mathscr{H}) \geq u_A(\mathscr{C}, \mathscr{H})$, then given that agent 2 is going to be computing honestly, agent 1 will prefer to do the same too, and due to symmetry, likewise for agent 2. In the game theoretic lingo, this means that $(\mathscr{H}, \mathscr{H})$ is a (Nash) equilibrium. If, further, $u_A(\mathscr{H}, \mathscr{C}) \geq u_A(\mathscr{C}, \mathscr{C})$, then $(\mathscr{H}, \mathscr{H})$ will be the dominant (Nash) equilibrium, i.e., honest computation is the preferred action irrespective of the action of the other agent.

The principal utilizes the redundancy scheme with probability α or employs only one of the agents (selected equally likely)[7] with probability $1 - \alpha$. If the principal chooses only one agent, then it audits it with probability ρ. Since auditing only occurs when a single agent receives the task, the likelihood λ that the task will ever be audited is $\rho(1-\alpha)$. As in the single-agent single-reward scenario, if only one agent is selected, the agent is rewarded r if there is no indication of wrongdoing, and is punished f if audited and caught wrong. When the redundancy scheme is selected and the returned results are equal, both agents are rewarded r. Otherwise, the principal performs an independent audit with probability β. When the results are different but the auditing does not occur, the principal has no way of identifying a culprit and treats them equally: both are fined at f. When the results are different and the auditing is done, then the cheating agent(s) are fined at f, but if one of the agents is right, then that agent is rewarded at the maximum value: R.[8] With the model so described, the expected utilities of an agent are computed as follows:[9]

$$u_A(\mathscr{H}, \mathscr{H}) = r - c, \qquad u_A(\mathscr{C}, \mathscr{H}) = (1 - \alpha - \lambda)r/2 - (\alpha + \lambda/2)f.$$

Hence, the condition $u_A(\mathscr{H}, \mathscr{H}) \geq u_A(\mathscr{C}, \mathscr{H})$ becomes: $r \geq (1 + \alpha)c/(\lambda + 2\alpha) - f$. Subject to making $(\mathscr{H}, \mathscr{H})$ an equilibrium, the contract is accepted if the expected utility of it to the agents is above their reserve utility, which we assume here too to be zero for simplicity: $r - c \geq 0$. Then the expected cost of the contract to the principal is:

$$\mathcal{C} = 2r\alpha + \gamma\lambda + r(1 - \alpha) = (1 + \alpha)r + \gamma\lambda.$$

The principal chooses λ, α, f, r, β such that honest computation is enforced, the contract is accepted, and the expected cost of the principal is minimized. λ and α must satisfy the structural condition $0 \leq \alpha \leq 1$, $0 \leq \lambda \leq \Lambda$ and $\alpha + \lambda \leq 1$. The instantaneous budget of the principal imposes $r \leq R$ if $\alpha = 0$, and $2r \leq R$ if $\alpha > 0$. We assume $R \geq 2c$, since otherwise, the principal can never employ both of the agents without violating its instantaneous budget constraint, and hence, the problem reduces to the

[7] We will formally show through the proof of proposition 5.3 that equal randomization is the best option. Intuitively, this removes any information that the agents may infer upon receiving a task.

[8] Note that this extra auditing threat of $\beta > 0$ is only instilled to make $(\mathscr{C}, \mathscr{C})$ (both agents cheating) a non-equilibrium if the goal is to make $(\mathscr{H}, \mathscr{H})$ the dominant Nash.

[9] Since the only information state to an agent is whether it receives the job, the ex-ante and ex-post analysis, i.e., before and after reception of the task, become equivalent. We present the ex-ante view for simplicity.

single agent problem. Then, the budget constraint simplifies to $r \leq R/2$. Therefore, the optimal contracts for two agents that make $(\mathscr{H}, \mathscr{H})$ an equilibrium are solutions of the optimization problem of:

$$\min_{r,f,\alpha,\lambda} r(1+\alpha) + \gamma\lambda \text{ subject to:}$$

$$r \leq R/2,\ f \leq F,\ 0 \leq \lambda \leq \Lambda,\ \lambda \leq 1-\alpha,\ \alpha \geq 0,\ r \geq c,\ r \geq \frac{c(1+\alpha)}{\lambda + 2\alpha} - f.$$

To make $(\mathscr{H}, \mathscr{H})$ the (unique) dominant Nash equilibrium, the principal should also ensure: $u(\mathscr{H}, \mathscr{C}) \geq u(\mathscr{C}, \mathscr{C})$, where:

$$u(\mathscr{H}, \mathscr{C}) = (1-\alpha)r/2 + \alpha\beta R - \alpha(1-\beta)f - c, \qquad u(\mathscr{C}, \mathscr{C}) = (1-\alpha-\lambda)r/2 - (\alpha+\lambda/2)f.$$

If the principal manages to make $(\mathscr{H}, \mathscr{H})$ a dominant Nash equilibrium, then the value of β does not directly affect the expected cost of contract. However, for the threat of the auditing when the results are dissimilar to be credible, the principal must have enough free auditing capacity left. Hence, we must have: $\lambda + \alpha\beta \leq \Lambda$. To attain the lowest cost contracts, the principal should set $\alpha\beta$ at its minimum possible value. $u(\mathscr{H}, \mathscr{C}) \geq u(\mathscr{C}, \mathscr{C})$ implies: $\alpha\beta \geq [c(1+\alpha) - \lambda(r+f)]/(2R+2f)$.

PROPOSITION 5.1. *Let $F_0 = c/\Lambda - c$ and $F_1 = c[c-\gamma]^+/[2\gamma - c]^+$,[10] the optimal one-level reward two-agent contract that makes $(\mathscr{H}, \mathscr{H})$ a Nash equilibrium is:*

$$\begin{cases} F_1 \leq F: & f^* = F,\ \alpha^* = \dfrac{c}{2F+c},\ \lambda^* = 0, r^* = c,\ \mathcal{C}^* = c(1 + \dfrac{c}{2F+c}) \\ F_0 \leq F < F_1: & f^* = F,\ \alpha^* = 0,\ \lambda^* = \dfrac{c}{c+F},\ r^* = c,\ \mathcal{C}^* = c(1 + \dfrac{\gamma}{F+c}) \\ F < \min(F_0, F_1): & f^* = F,\ \alpha^* = \dfrac{c - \Lambda(c+F)}{c+2F},\ \lambda^* = \Lambda,\ r^* = c,\ \mathcal{C}^* = \dfrac{c(c+F)(2-\Lambda)}{c+2F} + \gamma\Lambda \end{cases}$$

For $\Lambda = 1$, $(\mathscr{H}, \mathscr{H})$ is moreover the dominant Nash equilibrium.

COROLLARY 5.2. *If auditing is more expensive than the cost of honest computation ($\gamma \geq c$), the optimal contract only uses the redundancy scheme. When $\gamma \leq c/2$, either there is no redundancy scheme ($\alpha = 0$) or the whole auditing capacity is used ($\lambda^* = \Lambda$).*

The first part of the corollary is quite intuitive: when $\gamma > c$, any instance of outsourcing to a single agent and performing independent auditing can be replaced by the redundancy scheme (job duplication) and strictly lower the cost by $\gamma - c$.

Further Discussion. First, note that in our optimal two-agent contract, as long as $R \geq 2c$, there is no infeasible region: there is always a contract that makes $(\mathscr{H}, \mathscr{H})$ an equilibrium. Moreover, the payment to any of the agents is never more than the cost of honest computation. Fig. 3a provides a pictorial representation of the proposition where $c/2 < \gamma < c$ and $\Lambda = 0.5$. When the enforceable fine is large, the redundancy scheme is preferable. This is despite the fact that the redundancy scheme is more expensive than auditing: it costs an extra c as opposed to $\gamma < c$. In other words, for high values of fine, the redundancy scheme is a more effective threat against cheating than independent auditing. When F is less than F_1, the independent auditing becomes the preferred method. For lower values of F, when the auditing capacity is all used up, the redundancy scheme is added to compensate for the low value of fine to maintain incentive compatibility. Fig. 3b depicts the effect of auditing capacity, Λ, on the optimal contract where $c/2 < \gamma < c$. When $\Lambda = 0$, redundancy scheme is the only means to enforce honest computation. If furthermore no fine can be enforced ($F = 0$), then $\alpha = 1$:

[10] We adopt the convention that $x/0 = +\infty$ for $x > 0$.

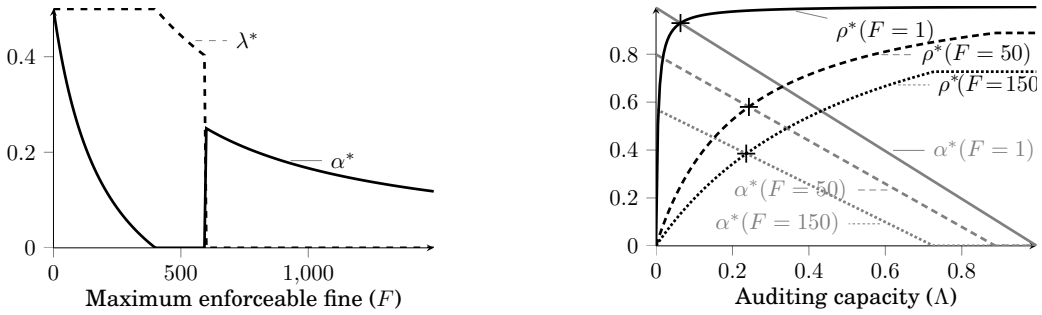

Fig. 3: Optimal contract (where $c = 400, \gamma = 250$) w.r.t. (a) max. enforceable fine F ($\Lambda = 0.5$); and (b) auditing capacity Λ ($F_1 = 600$). Recall $\rho = \frac{\lambda}{1-\alpha}$ is the conditional probability of auditing given the job is assigned to a single agent.

the job should be always duplicated. As Λ increases, there is a gradual transition from using redundancy scheme to independent auditing ($F < F_1$).

5.2. Global Optimality of Two-Agent Contracts

In developing the optimal contracts for two-agent case, we made a few critical assumptions: (a) the independent auditing is perfect; (b) the agents are non-colluding and non-communicating; (c) the range of intermediate steps is large enough that the probability of any two guessed results to be same, or the guessed result to be the correct result, is negligible; and (d) the agents are lazy but not malicious. It turns out that these assumptions are sufficient to warrant global optimality of two-agent contracts among contracts that engage any number of agents in the following notion:

PROPOSITION 5.3. *The contract that hires at most two agents and chooses its terms according to proposition 5.1, is globally optimal, that is, it achieves the least cost to the principal among all contracts that employ any number of agents and aim to make honest computation a Nash Equilibrium.*

The proof is provided in Appendix-D. The above proposition shows that our contract for two agents is not just a special case solution of multiple agents, but it is indeed the solution involving any number of agents. In other words, given the stipulated assumptions, there is no advantage ever in hiring more than two agents. Incidentally, the proof also shows that the best contracts makes the probability of any of the agents to be hired equal. This makes intuitive sense, as unequal probability of task assignment creates some "information" which the agents can potentially exploit to their benefit, and to the detriment of the principal.

6. CONTRACT IMPLEMENTATION

For completeness of the solutions, in this section we discuss notable technical concerns on the implementation of our contracts.

6.1. Intermediate steps and hash functions

As we discussed in Section 4.3, the use of intermediate steps as part of the output would prevent trivial/clever guessing. However, the data representing intermediate steps could be large and thus cumbersome for transmission. [Belenkiy et al. 2008] proposes the use of cryptographic hash as a sufficient representation of intermediate

steps: Instead of sending a large amount of data detailing these steps, the agent can only send the cryptographic hash of such data. On receiving the agent's hash h_A, the principal repeats the computation, and computes its own hash h_P from the intermediate steps, then verifies that $h_A = h_P$.

Informally, the use of hash function is considered secure if it is unlikely that the agent can come up with the correct hash without knowing the correct intermediate steps. The authors in [Belenkiy et al. 2008] require such hash function to be a "random oracle", i.e., a function mapping in which each output is chosen uniformly randomly regardless of the input. While this is a sufficient condition, the notion of random oracle is rather impractical, and also an overkill. Indeed, we argue that for this purpose of hash checking, it is necessary and sufficient that the hash function is either "collision resistant" or "one-way", that is, it should be difficult to either find two different messages with the same hash, or find a message that corresponds to a particularly chosen hash value.

Lastly, note that the process of hashing the intermediate steps may itself carry a considerable cost as well. For instance, if the computation task itself is to hash a large string, then the cost of hashing the intermediate steps (if the same hash function is used) would be at least as much as computation cost. Therefore, either the use of hash function on intermediate steps must be negligible compared to that of the original computation task, or the extra cost of hashing must enter the contract model.

6.2. Enforcing contract policies

With regards to legal enforcement of the contract, it is necessary that behaviours of contract participants are observable and verifiable. Actions such as "assigning a job" or "paying a reward" are of this type. However, probabilistic behaviours, e.g., "employing two agents with probability α", are usually unverifiable. Our contracts unfortunately rely on these probabilistic actions of the principal as explicitly stated in the terms and policies for auditing, task duplication and/or rewarding (the latter in two-level reward contracts of §4.5). It is critical to ensure (by both ends) that the principal in reality sticks to such actions, for two reasons: Firstly, the principal must establish to the agents its compliance to the contract so as to make the threats credible. Secondly, the agent needs an assurance that the principal cannot deviate from the contract and thus take away some of its benefits (in two-level rewarding). Without an appropriate security measure, this is usually not possible, e.g., the fact that the principal does not audit tells little about whether its auditing probability is indeed $\lambda = 0.3$ or $\lambda = 0.6$. This important contract implementation issue has not been discussed in previous works.

In Fig. 4, we propose a communication protocol between the principal and two agents that resolves this problem. Particularly, our security objective is to make sure that the principal gains negligible benefit by deviating from its prescribed behaviour as stated in the contract. To fulfil this objective, we rely on the fact that the contract can be legally enforced by an authority (e.g., a court), and thus punishment on the principal's cheating is guaranteed if there is enough evidence for the accusation. What remains is to ensure that agents can prove the principal's deviation (from the contract) whenever the principal benefits from doing so. Note that the principal may still deviate without additional benefit, which might be difficult to discover.

The main idea behind the protocol is to engage the agents into influencing the principal's probabilistic actions, as opposed to letting the principal to decide its actions alone. We allow each agent i to initiate a tuple of uniformly random nonces $\langle N_i, N_i^{(\alpha)}, N_i^{(\beta)}, N_i^{(\lambda)} \rangle$, each of which seeds a probabilistic part of the principal's action.

[11] Here the worst possible cost (including what has been spent) is $\max(2r, r + \gamma)$, and it could either be distributed to the agents, or paid to the court as fine.

Protocol ContractProtocol

Requirement. A (non-interactive) commitment scheme (Setup, Commit, Open) and a trusted third party TTP. An optimal contract $\langle r, f, \alpha, \beta, \lambda \rangle$, a computation task J, and a security parameter $k > 0$.

Preparation. The contract is signed by all parties with an additional term: if the principal is caught deviating from the below protocol, it must pay the worst possible cost [11]. TTP generates CK from Setup(k) and gives it to the principal (P), agent 1 (A_1) and agent 2 (A_2).

Protocol. (1) A_1: generates $N_1, N_1^{(\alpha)}, N_1^{(\beta)}, N_1^{(\lambda)} \leftarrow_r \{0,1\}^k$, computes $(c_1^*, d_1^*) =$ Commit$_{\mathsf{CK}}(\ N_1 || N_1^{(\alpha)} || N_1^{(\beta)} || N_1^{(\lambda)})$, then sends c_1^* to P.

(2) A_2: generates $N_2, N_2^{(\alpha)}, N_2^{(\beta)}, N_2^{(\lambda)} \leftarrow_r \{0,1\}^k$, computes $(c_2^*, d_2^*) =$ Commit$_{\mathsf{CK}}(\ N_2 || N_2^{(\alpha)} || N_2^{(\beta)} || N_2^{(\lambda)})$, then sends c_2^* to P.

(3) P: sends (c_1^*, c_2^*) to both A_1 and A_2.

(4) A_1: sends d_1^* to P.

(5) A_2: sends d_2^* to P.

(6) P: if $\frac{N_1^{(\alpha)} \oplus N_2^{(\alpha)}}{2^k} \leq \alpha$, then the job is sent to both, otherwise, it is sent to agent A_i, where $i = (N_1 \oplus N_2) (\text{mod } 2) + 1$. Results are sent back accordingly.

(7) P: if only one agent is hired, then the result is audited if $\frac{N_1^{(\lambda)} \oplus N_2^{(\lambda)}}{2^k} \leq \frac{\lambda}{1-\alpha}$. If two agents are hired, then when results are different, they are audited if $\frac{N_1^{(\beta)} \oplus N_2^{(\beta)}}{2^k} \leq \beta$.

(8) P: sends (d_1^*, d_2^*) to both A_1 and A_2.

Fig. 4: Communication protocol for the contract

For example, $N_i^{(\lambda)}$ are used to decide whether the principal will audit. The corresponding nonces from the two tuples are xor-ed together to preserve uniform randomness, in case one of the agents does not generate the nonces properly. Also, since we assume no collusion, agents do not communicate directly, and hence must give the nonces to the principal. Therefore, the only way for the agents to be sure that the principal treats the nonces properly is for them to also obtain these nonces (via the principal) from each other, and contrast against the observed principal's action. Using cryptographic commitments, we make sure that the principal cannot meaningfully modify agents' nonces before forwarding them, i.e., so that it can later deviate without being detected. After the computation results are returned, the principal securely reveals to the agents the behaviour it was expected to follow, then proceeds to rewarding/punishing the agents based on the returned results.

In addition, extra measures must be employed to ensure that legal enforcement is possible, i.e., all evidences must be verifiable. This can be accomplished with a signature scheme (KeyGen, Sign, Verify) which is *existentially unforgeable under chosen message attacks*. Each participant in the contract will be assigned (by a trusted third party) a public-private key pair generated using KeyGen, and must use its private key to sign every one of its sending messages. That way, supplying the conversation along with its signatures is sufficient for legal purposes. Our protocol can easily be adapted for single-agent contracts, in which the principal also plays the role of one agent. The

Table II: Examples of cost model for verification techniques

	recomp.	Yao's[Gennaro et al. 2010]	PEPPER[Setty et al. 2012a]	Traces[Monrose et al. 1999]	Quin[Canetti et al. 2011]
Principal	$\|f\|$	$\|f()\| \cdot \|\mathsf{Enc}_{full}\|$	$\|\pi(f)\| \cdot \|\mathsf{Enc}_{add}\|$	$\|f\|$	$\log \|f\|$
Agent(s)	$\|f\|$	$\|\mathsf{BC}(f)\| \cdot \|\mathsf{Enc}_{full}\|$	$\|\pi(f)\| \cdot \|\mathsf{Enc}_{add}\|$	$\|f\|$	$2\|f\|$

$|\ |$: size of the problem
$f()$: output of f
BC: boolean-circuit representation
π: PCP representation
Enc_{full} (resp. Enc_{add}): An operation of fully (resp. additive) homomorphic encryption

correctness and security of the protocol are informally stated as follows, which are formalised and proved in the appendix.

PROPOSITION 6.1. *Suppose that ContractProtocol is followed honestly by all participants, then the principal's behaviour is negligibly different from the contract's specification, i.e.,* $\langle r, f, \alpha, \beta, \lambda \rangle$.

PROPOSITION 6.2. *Suppose all participants in ContractProtocol are PPT algorithms. Suppose that (Setup, Commit, Open) is a secure non-malleable commitment scheme, and that contract terms can be legally enforced and that both agents are honest, then for any PPT principal that deviates from ContractProtocol, it receives additionally negligible payoff, given that the job is assigned to at least one agent.*

6.3. Modelling auditing/verification techniques

We mention from the beginning that our optimal contract does not replace rigorous verification techniques, but in fact relies on them as abstract auditing tools. It thus serves as a framework to not just study the economics of these techniques, but also compare and contrast among them. However, to fit our model, it is essential that the decision to audit must be hidden from the agent(s). For example, consider auditing using the scheme in [Gennaro et al. 2010]. Because it requires some preparation before the computation, the agent would know in advance whether an audit will happen, and thus probabilistic auditing is meaningless. As a workaround, we assume that this scheme is always active, but the principal would decide (with probability λ) whether to invoke the Verify step which costs $\mathcal{O}(m \cdot \mathsf{poly}(k))$, where m is the size of the Boolean circuit representing the computation, and poly(k) is some polynomial in a security parameter $k > 0$. That way the agent would be oblivious to whether the audit will happen. For a comparison, we have listed in Table II the cost model of some of the general-purpose verification methods in our literature review. It suggests that re-computation and redundancy are still enormously less expensive than notable cryptographic solutions.

7. CONCLUSION

In this paper, we provided an incentive-analysis of outsourced computation involving non-malicious but selfish utility-maximizing computing agents. We designed contracts that provably minimize the expected cost of the outsourcer while ensuring that computing parties accept the contracts and return correct results. We incorporated important featured inspired by real world restrictions, notably, that the outsourcing party can only levy a restricted fine on dishonest agents – since the solution becomes trivial if unrealistic, unlimited fines are allowed – and that auditing can be costly and/or limited. We allowed partial outsourcing, direct auditing and auditing through redundancy, i.e., employing multiple agents and comparing the results, and optimized the utility of the outsourcer among all hybrid possibilities.

We observed that outsourcing all or none of the tasks is optimal (and not partial outsourcing). We showed that when the enforceable fine is restricted, achieving honest computation may still be feasible by appropriately increasing the reward above the sheer cost of honest computation. We demonstrated that when auditing is more expensive than the cost of honest computation, redundancy scheme is always the preferred method, and when the auditing cost is less than half of the cost of honest computation, independent auditing is preferable. When the cost of auditing is between half and the full cost of honest computation, the preferred method depends on the maximum enforceable fine: for large enforceable fines, redundancy scheme is preferred despite the fact that it is more expensive "per use" than independent auditing, since owing to its higher effectiveness, it can be used more sparingly. We established the global optimality of contracts involving at most two agents among any arbitrary number of agents as far as implementing honesty as a Nash Equilibrium is aimed for. Finally, we presented a light-weight cryptographic implementation of our contracts that provided mutual affirmation on proper execution of the agreed terms and conditions.

An interesting problem for future research is dynamic contracts that either learn the hidden parameters of the utilities of agents or re-act to global changes in the computation costs and re-adjust the terms of contracts accordingly. Designing optimal contracts that are also resilient against collusions and/or communication between agents is also an interesting subject of future investigation.

REFERENCES

ATALLAH, M. J., CHO, Y., AND KUNDU, A. 2008. Efficient data authentication in an environment of untrusted third-party distributors. In *IEEE ICDE*.

BAZARAA, M. S., SHERALI, H. D., AND SHETTY, C. M. 2013. *Nonlinear programming: theory and algorithms*. John Wiley & Sons.

BELENKIY, M., CHASE, M., ERWAY, C. C., JANNOTTI, J., KÜPÇÜ, A., AND LYSYANSKAYA, A. 2008. Incentivizing outsourced computation. In *NetEcon*. ACM.

CANETTI, R., RIVA, B., AND ROTHBLUM, G. N. 2011. Practical delegation of computation using multiple servers. In *ACM CCS*.

CARBUNAR, B. AND TRIPUNITARA, M. V. 2012. Payments for outsourced computations. *IEEE Transactions on Parallel and Distributed Systems 23,* 2.

CHEN, H., MA, X., HSU, W., LI, N., AND WANG, Q. 2008. Access control friendly query verification for outsourced data publishing. In *ESORICS*.

CHRISTOFOROU, E., ANTA, A. F., GEORGIOU, C., MOSTEIRO, M. A., AND SÁNCHEZ, A. 2013. Applying the dynamics of evolution to achieve reliability in master–worker computing. *Concurrency and Computation: Practice and Experience*.

FULLBRIGHT, N. R. 2011. Outsourcing in a brave new world: An international survey of current outsourcing practice and trends. Tech. rep.

GENNARO, R., GENTRY, C., AND PARNO, B. 2010. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *CRYPTO*. Springer.

GINTIS, H. 2009. *Game Theory Evolving: A Problem-Centered Introduction to Modeling Strategic Interaction*. Princeton University Press.

MONROSE, F., WYCKOFF, P., AND RUBIN, A. D. 1999. Distributed execution with remote audit. In *NDSS*.

NIX, R. AND KANTARCIOGLU, M. 2012. Contractual agreement design for enforcing honesty in cloud outsourcing. In *GameSec*. Springer.

RASMUSEN, E. 1994. Games and information: an introduction to game theory.

SETTY, S., MCPHERSON, R., BLUMBERG, A. J., AND WALFISH, M. 2012a. Making argument systems for outsourced computation practical (sometimes). In *NDSS*.

SETTY, S., VU, V., PANPALIA, N., BRAUN, B., BLUMBERG, A. J., AND WALFISH, M.

2012b. Taking proof-based verified computation a few steps closer to practicality. In *USENIX Security*.

WANG, C., REN, K., AND WANG, J. 2011. Secure and practical outsourcing of linear programming in cloud computing. In *INFOCOM, 2011*.

YI, K., LI, F., CORMODE, G., HADJIELEFTHERIOU, M., KOLLIOS, G., AND SRIVASTAVA, D. 2009. Small synopses for group-by query verification on outsourced data streams. *ACM TODS*.

APPENDIX

A. PROOF OF PROPOSITION 4.1

We present the proof for the case of $\gamma > 0$. The case of $\gamma = 0$ follows more simply. For simplicity, let us fix a feasible fine and compute the solution in terms of f. In the end, we will show that $f = F$ is indeed optimal.[12] We will ignore the constraint of $\lambda \geq 0$ since it is strictly implied by the constraints in (3c) (the incentive compatibility). Furthermore, $r \geq H$ implies $r > 0$.

We use the Karush-Kuhn-Tucker (KKT) conditions [Bazaraa et al. 2013] to solve the above nonlinear (non-convex) programming.[13] Note that our cost and constraint functions are all continuously differentiable. We first use the Mangasarian–Fromovitz constraint qualification (MFCQ) to establish that any minimum must satisfy the KKT conditions, i.e., KKT are necessary conditions of optimality. In the absence of equality constraints, the MFCQ condition means that the gradients of the active inequality constraints are positive-linearly independent at optimum points.

In the special case of $R = H$, the constraints $r \geq H$ and $r \leq R$ imply $r = H = R$, and hence, the optimization problem can be rewritten with only λ as a variable, which is simple to analyse. When $R > H$, only one of the constraints $r - R \leq 0$ and $H - r \leq 0$ is ever active. We will investigate them one at a time. The gradients of the other inequality constraints $r - R \leq 0$, $\lambda - \Lambda \leq 0$ and $K - r\lambda - f\lambda \leq 0$ are respectively: $(1, 0)$, $(0, 1)$ and $(-\lambda, -r - f)$. Note that only for $f = K/\Lambda - R$, the last three inequalities can be all active. For this case, the domain of feasible solutions reduces to the singleton point of $r = R$, $\lambda = \Lambda$. For $f < K/\Lambda - R$, no feasible solution exists. For all other cases, at most two of the constraints are active at a time, whose gradients can never be linearly dependent: $(1, 0)$ and $(0, 1)$ are clearly linearly independent, and both elements of $(-\lambda, -r - f)$ are strictly negative, hence it is linearly independent from each of the other two. Now, consider the $H - r \leq 0$ constraint whose gradient is $(-1, 0)$. Note that three of the constraints may be simultaneously active, but their gradient will not be positive-linearly dependent, because both elements of $(-\lambda, -r-f)$ are strictly negative. Hence, the MFCQ normality condition holds.

To systematically obtain the KKT conditions, we introduce the dual multipliers μ_1, μ_2, μ_3 and μ_4, and transform the problem in (3) as follows:

$$\min_{r,f,\lambda,\mu_i} \bar{\mathcal{C}} = r + \gamma\lambda + \mu_1(r - R) + \mu_2(\lambda - \Lambda) + \mu_3(H - r) + \mu_4(K - f\lambda - r\lambda)$$

$$\text{s.t.:} \quad \text{primary feasibility: } r \leq R,\ \lambda \leq \Lambda,\ r \geq H,\ r\lambda + f\lambda \geq K \tag{7a}$$

$$\text{duality feasibility: } \mu_1, \mu_2, \mu_3, \mu_4 \geq 0, \tag{7b}$$

$$\text{complementary slackness: } \mu_1(r - R) = 0,\ \mu_2(\lambda - \Lambda) = 0, \tag{7c}$$

[12] Alternatively, the following simple argument shows from the beginning that f must be at its maximum value F: Note that the principal can increase the auditing rate λ, or reward r or the fine f in order to enforce the incentive compatibility constraint. Of these three variables, only increasing the fine is costless to the principal.

[13] The nonconvexity arises due to the second inequality in (3c).

$$\mu_3(H-r)=0,\ \mu_4(K-f\lambda-r\lambda)=0. \tag{7d}$$

The first order conditions of optimality are:

$$\frac{\partial\bar{\mathcal{C}}}{\partial r}=0\Leftrightarrow\mu_4\lambda=1+\mu_1-\mu_3,\qquad \frac{\partial\bar{\mathcal{C}}}{\partial\lambda}=0\Leftrightarrow\mu_4 r=\gamma+\mu_2-f\mu_4. \tag{8}$$

The full solution as in the proposition with F replaced by f is now derived by straightforward investigation of the above conditions. The proof then concludes by noting that the cost such found is strictly decreasing in f, and hence $f^*=F$.

B. PROOF OF PROPOSITION 4.2

The only two constraints in the optimal contract that may change are the incentive compatibility and participation: (1), (2). The new participation constraint is: $u(r-c)\geq u(z)$. Due to the increasing property of $u(\cdot)$, this new constraints translates back to $r-c\geq z$, hence no change here.

For analysing the new incentive-compatibility constraint, let us represent the mixed action of the agent by $\mathcal{L}(x,y,1-x-y)$ which means making a random guess with probability x, using the tricky algorithm with probability y, and doing the honest computation with probability $1-x-y$. With a slight abuse of notation, let $X[\mathcal{L}]$ be the random variable representing the utility of the agent given its mixed action $\mathcal{L}$. Then the risk-neutral incentive compatibility constraint as given in (1) is ensuring that $\mathbb{E}X[\mathcal{L}(0,0,1)]\geq\mathbb{E}X[\mathcal{L}(x,y,1-x-y)]$. Because $u(\cdot)$ is increasing, this inequality implies: $u\left(\mathbb{E}X[\mathcal{L}(0,0,1)]\right)\geq u\left(\mathbb{E}X[\mathcal{L}(x,y,1-x-y)]\right)$. Further, following Jensen's inequality, since u is concave, $u\left(\mathbb{E}X[\mathcal{L}(x,y,1-x-y)]\right)\geq\mathbb{E}u\left(X[\mathcal{L}(x,y,1-x-y)]\right)$. Note that $X[\mathcal{L}(0,0,1)]$ is a deterministic random variable (specifically, payoff of $r-c(1)$ w.p. one). Hence: $u\left(\mathbb{E}X[\mathcal{L}(x,y,1-x-y)]\right)=\mathbb{E}u\left(X[\mathcal{L}(x,y,1-x-y)]\right)$. Therefore, we have shown that (1) implies: $\mathbb{E}u\left(X[\mathcal{L}(x,y,1-x-y)]\right)\geq\mathbb{E}u\left(X[\mathcal{L}(x,y,1-x-y)]\right)$, which is the incentive compatibility constraint for a risk-averse agent.

C. PROOF OF PROPOSITION 4.3

Let y represent the probability that the task is outsourced to the agent *and* the principal audits it. For a general choice of $x\in[0,\Lambda]$, the expected cost of the principal is $\mathcal{C}=r(1-x)+\gamma(x+y)$. The new incentive compatibility for honest computation (given the agent receives the task) is: $r-c\geq\frac{r(1-x-y)}{1-x}-\frac{fy}{1-x}\Leftrightarrow r\geq\frac{c(1-x)}{y}-f$. An optimal contract is thus a solution of $\min_{r,f,x,y}\mathcal{C}:=r(1-x)+\gamma(x+y)$ subject to: $r\leq R,\ f\leq F,\ 0\leq x,y\leq\Lambda,\ x+y\leq\Lambda,\ r\geq c,\ r\geq\frac{c(1-x)}{y}-f$. The rest of the proof follows similar to that of Proposition 4.1.

D. PROOF OF PROPOSITION 5.3

PROOF. First, we provide an argument that if the optimal contract indeed assigns a task to more than one agent, then it does not benefit from hiring them agents in a "sequential" manner. Next, we prove that two-agent contract is the best solution among all "non-sequential" contracts by converting any given contract to a two-agent contract and improving the cost of the outsourcer.

Suppose that the optimal contract hired the agents sequentially for a given task. If in the first step, more than one agent is hired, then there are two possibilities: either (a) the returned results are the same, in which case, the computation is correct and there is no point in hiring any more agents and any subsequent steps; or (b) the returned results are different, which means that at least one of the agents has not computed the task correctly, in which case, all of the agents can be punished which includes the wrongdoer, and hence there is no need for any subsequent steps. If in the

first step only one agent is assigned the task, there are again two possibilities: (a) the agent is audited, in which case the principal unequivocally knows whether cheating has occurred and hence, there is no need for subsequent steps; or (b) the returned result is not audited. In the latter case, the agents that are hired in the next immediate step can be combined with the single agent hired in the first step and thought of as they are all hired in a the step. Then, the argument for multiple agents in the first step can be applied to remove the need for any subsequent steps. Therefore, any optimal contract is either non-sequential or can be converted to a non-sequential one.

Now, let λ_j be the (unconditional) probability that agent j is independently audited. Also let p_{ji} be the (conditional) probability that agent j receives the task if i agents are assigned. The expected cost of the contract to the principal is the following:

$$\mathcal{C} = r\sum_{i=2}^{N} i\alpha_i + r(1-\sum_{i=1}^{N}\alpha_i) + \gamma\sum_{j=1}^{N}\lambda_j = r\sum_{i=2}^{N}(i-1)\alpha_i + r + \gamma\sum_{j=1}^{N}\lambda_j$$

Let φ_j be the probability that agent j receives the task. Then: $\varphi_j = \sum_{i=2}^{N} p_{ji}\alpha_i + p_{j1}(1-\sum_{i=2}^{N}\alpha_i)$. The expected utility of agent j for honest computation given that it has received the message, and given that the rest of the agents are honest, is simply $r-c$. Now, let us define ψ_j to be the probability that agent j is rewarded given its strategy is to cheat. Given the honesty of all other agents, agent j is rewarded only if it is the only one that is assigned the task and it is not audited on. This gives: $\psi_j = p_{j1}(1-\sum_{i=2}^{N}\alpha_i) - \lambda_j$. Therefore, the expected utility of agent j for cheating given it is assigned the task and all other agents are honest is $r\psi_j/\varphi_j - f(\varphi_j-\psi_j)/\varphi_j$. Hence, the incentive compatibility constraint, i.e., $u_j(\mathscr{H}, \overline{\mathscr{H}}) \geq u_j(\mathscr{C}, \overline{\mathscr{H}})$ with $\overline{\mathscr{H}}$ representing the honest strategy of other $N-1$ agents, becomes: $r \geq \dfrac{\varphi_j}{\varphi_j - \psi_j}c - f$. Taking into account the incentive compatibility of all of the agent, we have:

$$r \geq \max_j \frac{\sum_{i=2}^{N} p_{ji}\alpha_i + p_{j1}(1-\sum_{i=2}^{N}\alpha_i)}{\sum_{i=2}^{N} p_{ji}\alpha_i + \lambda_j}$$

The participation constraint given that the honesty of all agents is established is simply $r - c \geq 0$. Thus, the optimal contract is given by the following optimization:

$$\min_{\alpha_i,\lambda_j,p_{ji},r,f} \mathcal{C} = r\sum_{i=2}^{N}(i-1)\alpha_i + r + \gamma\sum_{j=1}^{N}\lambda_j$$

$$\text{s.t.} \quad Nr \leq R,\ f \leq F,\ \lambda_j, p_{j1}, \alpha_i \geq 0,\ \lambda_j \leq (1-\sum_{i=2}^{N}\alpha_i)p_{j1},\ \sum_{j=1}^{N} p_{ji} = 1\ \forall i,\ \sum_{j=1}^{N}\lambda_j + \sum_{i=2}^{N}\alpha_i \leq 1,$$

$$\sum_{j=1}^{N}\lambda_j \leq \Lambda,\ r \geq \max_j \frac{\sum_{i=2}^{N} p_{ji}\alpha_i + p_{j1}(1-\sum_{i=2}^{N}\alpha_i)}{\sum_{i=2}^{N} p_{ji}\alpha_i + \lambda_j},\ r - c \geq 0.$$

Now, suppose there is a claimed solution in which $\alpha_i > 0$ for at least one $i \in \{3, \ldots, N\}$. In what follows we construct an alternative solution that improves the cost to the principal in which $\alpha_i = 0$ for all $3 \leq i \leq N$. Consider this alternative contract:

$$\hat{\alpha}_2 = \sum_{i=2}^{N}\alpha_i,\ \hat{\alpha}_i = 0\ \forall i \geq 3,\ \hat{\lambda}_j = \sum_{k=1}^{N}\lambda_k/N, \hat{p}_{ji} = 1/N\ \forall i, j,\ \hat{r} = r,\ \hat{f} = f.$$

First, we show that given the feasibility of the claimed contract, this alternative contract is also feasible, and subsequently, establish the improvement in the achieved cost. The only non-trivial constraint to check for feasibility of the above contract is the incentive compatibility constraint:

$$r \geq \frac{\frac{1}{N}\sum_{i=2}^{N}\alpha_i + \frac{1}{N}(1-\sum_{i=2}^{N}\alpha_i)/N}{\frac{1}{N}\sum_{i=2}^{N}\alpha_i + \frac{1}{N}\sum_{j=1}^{N}\lambda_j} = \frac{1}{\sum_{i=2}^{N}\alpha_i + \sum_{j=1}^{N}\lambda_j} \tag{9}$$

From the feasibility of the claimed contract, we have:

$$r \geq \max_j \frac{\sum_{i=2}^{N} p_{ji}\alpha_i + p_{j1}(1-\sum_{i=2}^{N}\alpha_i)}{\sum_{i=2}^{N} p_{ji}\alpha_i + \lambda_j} \Rightarrow r \geq \frac{\sum_{j=1}^{N}\left(\sum_{i=2}^{N} p_{ji}\alpha_i + p_{j1}(1-\sum_{i=2}^{N}\alpha_i)\right)}{\sum_{j=1}^{N}\left(\sum_{i=2}^{N} p_{ji}\alpha_i + \lambda_j\right)}$$

$$= \frac{\left(\sum_{i=2}^{N}\sum_{j=1}^{N} p_{ji}\alpha_i + \sum_{j=1}^{N} p_{j1}(1-\sum_{i=2}^{N}\alpha_i)\right)}{\left(\sum_{i=2}^{N}\sum_{j=1}^{N} p_{ji}\alpha_i + \sum_{j=1}^{N}\lambda_j\right)} = \frac{\sum_{i=2}^{N}\alpha_i + (1-\sum_{i=2}^{N}\alpha_i)}{\sum_{i=2}^{N}\alpha_i + \sum_{j=1}^{N}\lambda_j}$$

which gives (9). This establishes that the new solution is also feasible. In the first line of the above argument, we used the following simple lemma:

LEMMA D.1. *If we have* $a \geq \max_{j\in\mathcal{J}} \frac{b_j}{c_j}$ *where* $c_j > 0$ *for all* $j \in \mathcal{J}$*, then* $a \geq \frac{\sum_{j\in\mathcal{J}} b_j}{\sum_{j\in\mathcal{J}} c_j}$.

Now: $\hat{\mathcal{C}} = \hat{r}\sum_{i=2}^{N}(i-1)\hat{\alpha}_i + \hat{r} + \gamma\hat{\lambda} = r\sum_{i=2}^{N}\alpha_i + r + \gamma\sum_{j=1}^{N}\lambda_j \leq r\sum_{i=2}^{N}(i-1)\alpha_i + r + \gamma\sum_{j=1}^{N}\lambda_j = \mathcal{C}$. □

E. PROOF OF PROPOSITION 6.1

PROOF. This result is rather straightforward. We notice that the principal's behaviour is driven by the nonces, as indicated in step 6 and 7 of ContractProtocol. Since the nonces are generated uniformly randomly from $\{0,1\}^k$, the xor-ed version of the nonces (e.g., $N_1^{(\alpha)} \oplus N_2^{(\alpha)}$) are also uniformly random. It is then easy to verify that the probabilistic behaviour of the principal is statistically indistinguishable from the contract specification. For example, considering the decision of whether one or two agents are employed, with X is the random variable in that $X = 2$ with probability α, then

$$|\Pr[\frac{N_1^{(\alpha)} \oplus N_2^{(\alpha)}}{2^k} \leq \alpha] - \Pr[X=2]| = |\Pr[\frac{N \leftarrow_r \{0,1\}^k}{2^k} \leq \alpha] - \alpha| = |\frac{\lfloor \alpha 2^k \rfloor}{2^k} - \alpha| \leq \frac{1}{2^k} \tag{10}$$

Other parts of the behaviour can be proved similarly. □

F. PROOF OF PROPOSITION 6.2

PROOF. Before proceeding with the proof, let us formalise the statement of the proposition, particularly focusing on the notions of "behaviour", "deviation", "caught". Firstly, the behaviour of the principal is characterised by how it plans to act (and will eventually do so) in implementing the contract. A *plan* of action for the principal essentially captures the principal's choices for all possible decision-making situations which might arise while executing the contract. For convenience we denote the set of all possible plans as Ω, which also contains an element $\perp$ representing an invalid plan. The principal P is supposed to pick a plan $\omega \in \Omega$ according a contract-specific probability distribution $\Delta(\Omega)$, but the agents do not know if P actually follows this distribution, or a different one to its eventual benefit.

As a result, we decide to let such a plan be picked by the agents rather than the principal. The protocol for "picking plan" should satisfy the following properties:

— *Correctness*: Honest execution of the protocol must ensure that the plan is picked according $\Delta(\Omega)$.
— *Hiding*: Before the contract is executed, the agents must know nothing about the picked plan.
— *Revealing*: After the contract is executed, there must be a secure way for the plan to be revealed to the agents.
— *No cheating*: Suppose that the agents execute the protocol honestly, then by deviating from it the principal does receive any better benefit.

While the need for correctness is clear, we note that the hiding property guarantees our original assumption that the agents do not know the principal's behaviour until they have carried out and returned the computation. Finally, the revealing and no cheating properties facilitate detection of the principal's deviation and/or assurance that such deviation is not beneficial. Informally, detection of deviation is a process in which an agent contrasts what it obverses as the principal's behaviour against a plan $\omega \in \Omega$ that was picked by the agents. An inconsistency between the plan and the observed behaviour would indicate to that agent of the principal's deviation.

Next, we define $\mathcal{B}$ to be the set of possible *observable* behaviours, along with *consistency relations* $\sim_1$ and $\sim_2$ (for A_1 and A_2 respectively), such that $\omega \sim_i b \Leftrightarrow b \sim_i \omega$ for all pairs $(\omega, b) \in \Omega \times \mathcal{B}$ and $i = 1, 2$. Here an observable behaviour is a collection of actions that could be easily and costlessly reproduced (in front of the judges) for verification. For example, the action of assigning a job to an agent, or giving a reward (shown in the bank statement) are observable.

Finally, in order to capture the principal's benefit from deviation, we need to define its utility for each outcome of the contract execution. Essentially this utility is a function $u_P : \Omega^2 \times (\mathcal{B}^2 \setminus \{(\emptyset, \emptyset)\}) \to \mathbb{R}$, in which a tuple $(\omega_1, \omega_2, b_1, b_2) \in \Omega^2 \times \mathcal{B}^2$ indicates the agents' views on the chosen plan (i.e., ω_1 and ω_2), and the principal's chosen behaviours regarding each agent (i.e., b_1 and b_2). Here $\emptyset$ denotes the fact that an agent is not employed, and hence $(\emptyset, \emptyset)$ should be excluded, since in that case the principal will need to form a new contract (possibly with different agents).

Our solution relies heavily on commitments, and hence can be described by a tuple $(\mathcal{S}, \mathcal{C}, \mathcal{D})$ of setup, commit, and decommit algorithms. Formal security definition of the above requirements is:

Definition F.1. A secure contract implementation mechanism for $\Delta(\Omega)$, $\mathcal{B}$, $\sim_1$, $\sim_2$, and $u_P : \Omega^2 \times (\mathcal{B}^2 \setminus \{(\emptyset, \emptyset)\}) \to \mathbb{R}$ is a tuple of PPT algorithms $(\mathcal{S}, \mathcal{C}, \mathcal{D})$ such that:

— **Correctness:** given the following experiments

$\mathbf{Prot}(\Delta(\Omega), k)$:	$\mathbf{Ideal}(\Delta(\Omega))$:
$\mathsf{CK} \leftarrow \mathcal{S}(k)$	$\omega \leftarrow \Delta(\Omega)$
$(c_1, d_1) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega))$	$b_1 \leftarrow_R \{b \in \mathcal{B} \mid b \sim_1 \omega\}$
$(c_2, d_2) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega))$	$b_2 \leftarrow_R \{b \in \mathcal{B} \mid b \sim_2 \omega\}$
$(\omega, b_1, b_2) \leftarrow \mathcal{D}_{\mathsf{CK}}(c_1, d_1, c_2, d_2)$	$\mathbf{return}(\omega, b_1, b_2)$
$\mathbf{return}(\omega, b_1, b_2)$	

it must hold that $\mathbf{Prot}(\Delta(\Omega), k)$ and $\mathbf{Ideal}(\Delta(\Omega))$ are statistically indistinguishable, i.e.,

$$\sup_{o \in \Omega \times \mathcal{B}^2} |\Pr[o \leftarrow \mathbf{Prot}(\Delta(\Omega), k)] - \Pr[o \leftarrow \mathbf{Ideal}(\Delta(\Omega))]| \leq \epsilon(k)$$

— **Hiding:** for every PPT algorithm $A = (A_1, A_2)$ (curious agent), there exists a PPT algorithm $A^* = (A_1^*, A_2^*)$ satisfying:

$$\Pr\left[\begin{array}{c}\mathsf{CK} \leftarrow \mathcal{S}(k); (c,d) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega)); (c',d',m) \leftarrow A_1(\Delta(\Omega), \mathsf{CK}); \omega \leftarrow \\ A_2(c,m,\Delta(\Omega),\mathsf{CK}); (\omega', b_1, b_2) \leftarrow \mathcal{D}_{\mathsf{CK}}(c,d,c',d') : \perp \neq \omega = \omega'\end{array}\right]$$
$$- \Pr\left[\begin{array}{c}\mathsf{CK} \leftarrow \mathcal{S}(k); (c,d) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega)); (c',d',m) \leftarrow A_1^*(\Delta(\Omega), \mathsf{CK}); \\ \omega^* \leftarrow A_2^*(m,\Delta(\Omega),\mathsf{CK}); (\omega', b_1, b_2) \leftarrow \mathcal{D}_{\mathsf{CK}}(c,d,c',d') : \perp \neq \omega^* = \omega'\end{array}\right] \leq \epsilon(k) \quad (11)$$

— **Revealing:** for every PPT algorithm P (cheating principal) we have

$$\Pr\left[\begin{array}{c}\mathsf{CK} \leftarrow \mathcal{S}(k); (c,d) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega)); (c',d_1,d_2) \leftarrow P(c,d,\Delta(\Omega),\mathsf{CK}); (\omega_1,\cdot,\cdot) \leftarrow \\ \mathcal{D}_{\mathsf{CK}}(c,d,c',d_1); (\omega_2,\cdot,\cdot) \leftarrow \mathcal{D}_{\mathsf{CK}}(c,d,c',d_2) : \omega_1 \neq \omega_2 \wedge \omega_1, \omega_2 \neq \perp\end{array}\right]^{14} \leq \epsilon(k) \quad (12)$$

— **No cheating:** given the following experiment

$\mathbf{Real}(\Delta(\Omega), k, P)$:
$\mathsf{CK} \leftarrow \mathcal{S}(k)$
$(c_1, d_1) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega))$
$(c_2, d_2) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega))$
$(c_1', c_2') \leftarrow P(c_1, c_2)$
if $c_1' = c_2 \vee c_2' = c_1$ **return** $(\perp, \perp, \emptyset, \emptyset)$[15]
$(d_1', d_2', b_1, b_2) \leftarrow P(c_1, d_1, c_2, d_2)$
$(\omega_1, \cdot, \cdot) \leftarrow \mathcal{D}_{\mathsf{CK}}(c_1, d_1, c_2', d_2')$
$(\omega_2, \cdot, \cdot) \leftarrow \mathcal{D}_{\mathsf{CK}}(c_1', d_1', c_2, d_2)$
return $(\omega_1, \omega_2, b_1, b_2)$

it must hold that for all PPT algorithm P (cheating principal)

$$\mathbb{E}[u_P(\omega_1,\omega_2,b_1,b_2) | (\omega_1,\omega_2,b_1,b_2) \leftarrow \mathbf{Real}(\Delta(\Omega),k,P); (b_1,b_2) \neq (\emptyset,\emptyset)]$$
$$- \mathbb{E}[u_P(\omega,\omega,b_1,b_2) | (\omega,b_1,b_2) \leftarrow \mathbf{Prot}(\Delta(\Omega),k); (b_1,b_2) \neq (\emptyset,\emptyset)] \leq \epsilon(k) \quad (13)$$

The next step before constructing a protocol that satisfies the above definition is to formalise Ω, $\mathcal{B}$, $\sim_1$, $\sim_2$, and u_P. The set of possible plans can be described as

$$\Omega = \{A_1, A_2\} \times \{audit, reward\} \cap \{(A_1, A_2, reward)\} \times \{audit, punish\ all\} \cap \{\perp\}.$$

In other words, a plan must specify the employed agent(s), as well as what to do when the results are returned. The $\{audit, punish\ all\}$ option for two-agent employment specifies the principal's action when conflicting results are returned. Also for convenience we put $reward$ alongside two-agent employment to indicate that when identical results are returned, the reward is blindly given to the agents. Note that in each plan we do not consider reward and punishment values, because these figures are set out clearly in the contract. As a result, we will also not consider them in the observable behaviour, as below:

$$\mathcal{B} = \{A_1, A_2\} \times \{audit, reward\} \cap \{\emptyset\}$$

Our construction of $\mathcal{B}$ relies on the *assumption* that the acts of auditing and blind rewarding are distinguishable and verifiable to the agents. Note that we do not consider

[14] The $\cdot$ symbol indicates an irrelevant output which we omit for simplification.
[15] This is to prevent the cheating principal P to return a commitment back to an agent, pretending that it comes from another one, as that would clearly nullify all the seeds for selecting $\omega_i \in \Omega$.

punish all because this cannot happen when both agents perform honest computation. Next, the consistency relations follow straightforwardly, i.e., for all $b \in \mathcal{B}$ and $\omega \in \Omega$:

$$b \sim_i \omega \iff \omega \sim_i b \iff \begin{cases} b = \emptyset \wedge \ A_i \text{ does not appear in } \omega \\ b \neq \emptyset \wedge \forall t \text{ appears in } b,\ t \text{ appears in } \omega \end{cases}$$

Evidently, such definition of consistency implies that for any $\omega \in \Omega$ and for any $i \in \{1, 2\}$, there is only one $b \in \mathcal{B}$ such that $b \sim_i \omega$. Finally, the principal's utility is captured in the following:

$$r_P(\omega_1, \omega_2, b_1, b_2) = (b_1 = \emptyset \ ?\ 0 : -r) + (b_2 = \emptyset \ ?\ 0 : -r)^{16} \tag{14}$$

$$c_P(\omega_1, \omega_2, b_1, b_2) = (audit \text{ appears in } b_1 \ ?\ \gamma : 0) + (audit \text{ appears in } b_2 \ ?\ \gamma : 0) \tag{15}$$

$$u_P(\omega_1, \omega_2, b_1, b_2) = \begin{cases} \min(-2r, -r - \gamma) \text{ if } b_1 \not\sim_1 \omega_1 \vee b_2 \not\sim_2 \omega_2, \\ r_P(\omega_1, \omega_2, b_1, b_2) + c_P(\omega_1, \omega_2, b_1, b_2) \text{ otherwise.} \end{cases} \tag{16}$$

To prove that ContractProtocol satisfies the above security definition with regard to Ω, $\mathcal{B}$, $\sim_1$, $\sim_2$, and u_P we first convert it into the tuple $(\mathcal{S}, \mathcal{C}, \mathcal{D})$, utilising the commitment scheme $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open})$:

proc $\mathcal{S}(k)$:
$\mathsf{CK} \leftarrow \mathsf{Setup}(k)$
return(CK)

proc $\mathcal{C}_{\mathsf{CK}}(\Delta(\Omega))$:
$\langle \alpha, \beta, \lambda \rangle \leftarrow \Delta(\Omega)$
$(N, N^{(\alpha)}, N^{(\beta)}, N^{(\alpha)}) \leftarrow_r \{0,1\}^k$
$(c, d) = \mathsf{Commit}_{\mathsf{CK}}(N||N^{(\alpha)}||N^{(\beta)}||N^{(\lambda)})$
return$(c||\langle \alpha, \beta, \lambda \rangle, d)$

proc $\mathcal{D}_{\mathsf{CK}}(c_1, d_1, c_2, d_2)$:
$c_i'||\langle \alpha_i, \beta_i, \lambda_i \rangle \leftarrow$ **parse**(c_i) for $i \in \{1, 2\}$
if $\langle \alpha_1, \beta_1, \lambda_1 \rangle \neq \langle \alpha_2, \beta_2, \lambda_2 \rangle$ **return** $(\perp, \emptyset, \emptyset)$
if $\mathsf{Open}_{\mathsf{CK}}(c_i', d_i) = \perp,\ i \in \{1, 2\}$ **return** $\{\perp, \emptyset, \emptyset\}$
$N_i||N_i^{(\alpha)}||N_i^{(\beta)}||N_i^{(\alpha)} \leftarrow \mathsf{Open}_{\mathsf{CK}}(c_i', d_i)$ for $i \in \{1, 2\}$
$a = (N_1 \oplus N_2) (\mathrm{mod}\ 2) = 0 \ ?\ A_1 \ :\ A_2$
$b = (N_1^{(\lambda)} \oplus N_2^{(\lambda)})/2^k \leq \lambda/(1-\alpha) \ ?\ audit \ :\ reward$
if $(N_1^{(\alpha)} \oplus N_2^{(\alpha)})/2^k \leq \alpha$:
  $a = (A_1, A_2, reward)$
  $b = (N_1^{(\beta)} \oplus N_2^{(\beta)})/2^k \leq \beta \ ?\ audit \ :\ punish\ all$
$b_i \leftarrow_R \{b \in \mathcal{B} : b \sim_i (a, b)\},$ for $i \in \{1, 2\}$
return$((a, b), b_1, b_2)$

We prove that this protocol satisfies our desired security properties:

(1) **Correctness.** This has been proved in Proposition 6.1.
(2) **Hiding.** We assume that there exists a PPT algorithm A such that no PPT algorithm A^* can preserve (11) and construct an adversary A' against the hiding property of the commitment scheme. First, given the existence of $A = (A_1, A_2)$, let $A^* = (A_1, A*_2)$ such that $A_2^*(m, \Delta(\Omega), \mathsf{CK}) = A_2(c||\langle \alpha, \beta, \gamma \rangle, m, \Delta(\Omega), \mathsf{CK})$, where $(c, d) \leftarrow \mathsf{Commit}_{\mathsf{CK}}(0^{4k})$. The fact that (11) does not hold thus implies

$$\Pr\left[\begin{array}{c} \mathsf{CK} \leftarrow \mathcal{S}(k); (c, d) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega)); (c', d', m) \leftarrow A_1(\Delta(\Omega), \mathsf{CK}); \omega \leftarrow \\ A_2(c, m, \Delta(\Omega), \mathsf{CK}); (\omega', b_1, b_2) \leftarrow \mathcal{D}_{\mathsf{CK}}(c, d, c', d') : \perp \neq \omega = \omega' \end{array}\right]$$

$$- \Pr\left[\begin{array}{c} \mathsf{CK} \leftarrow \mathcal{S}(k); (c, d) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega)); (c', d', m) \leftarrow A_1(\Delta(\Omega), \mathsf{CK}); (c^*, d^*) \leftarrow \mathsf{Commit}_{\mathsf{CK}}(0^{4k}); \\ \omega^* \leftarrow A_2(c^*||\langle \alpha, \beta, \gamma \rangle, m, \Delta(\Omega), \mathsf{CK}); (\omega', b_1, b_2) \leftarrow \mathcal{D}_{\mathsf{CK}}(c, d, c', d') : \perp \neq \omega^* = \omega' \end{array}\right]$$

$$> \epsilon \tag{17}$$

[16] The C-like operator $bool\ ?\ a : b$ gives value a if $bool =$ **true** and b otherwise.

As a result, the attacker A' against the commitment scheme proceeds as follows:

proc $AttackCommitHiding$:
$\mathsf{CK} \leftarrow \mathsf{Setup}(k)$
$(c, d) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega))$
$m_0^* \leftarrow \mathsf{Open}_{\mathsf{CK}}(c, d)$
$m_1 \leftarrow 0^{4k}$
$(c', d', m) \leftarrow A_1(\Delta(\Omega), \mathsf{CK})$
$(\omega, \cdot, \cdot) \leftarrow \mathcal{D}(c, d, c', d')$
$b \leftarrow_r \{0, 1\}$
$(c^*, d^*) \leftarrow \mathsf{Commit}_{\mathsf{CK}}(m_b)$
$\omega' \leftarrow A_2(c^*, m, \Delta(\Omega), \mathsf{CK})$
$b' = (\omega = \omega' \ ? \ 0 \ : \ 1)$
return $b = b'$

Denote (17) in short as $p - q > \epsilon$ we notice that when $b = 0$, A_2's view of $AttackCommitHiding$ experiment is identical to the experiment associated with p. Thus, in that case $\Pr[\omega = \omega'] = p$. Similarly, when $b = 1$ we have $\Pr[\omega \neq \omega'] = 1 - q$. Therefore,

$$\begin{aligned}
\Pr[AttackCommitHiding = \textbf{true}] &= \Pr[b = 0, b' = 0] + \Pr[b = 1, b' = 1] \\
&= \Pr[b = 0] \Pr[b' = 0 | b = 0] + \Pr[b = 1] \Pr[b' = 1 | b = 1] \\
&= \frac{1}{2} p + \frac{1}{2}(1 - q) \\
&> \frac{1}{2}(q + \epsilon) + \frac{1}{2}(1 - q) = \frac{1}{2} + \frac{1}{2}\epsilon
\end{aligned}$$

This thus proves that the hiding property of the commitment scheme is broken, which contradicts with our assumption on its security.

(3) **Revealing.** Similar to the previous proof, we show that an attacker P on this property can be used to construct an attacker P' on the *binding* property of the commitment scheme. The proof, however, is rather straightforward:

proc $AttackCommitBinding$:
$\mathsf{CK} \leftarrow \mathsf{Setup}(k)$
$(c || \langle \alpha, \beta, \lambda \rangle, d) \leftarrow \mathcal{C}_{\mathsf{CK}}(\Delta(\Omega))$
$(c' || \langle \alpha', \beta', \lambda' \rangle, d_1, d_2) \leftarrow P(c, \Delta(\Omega), \mathsf{CK})$
$m_1 \leftarrow \mathsf{Open}_{\mathsf{CK}}(c', d_1)$
$m_2 \leftarrow \mathsf{Open}_{\mathsf{CK}}(c', d_2)$
return $(m_1 \neq m_2 \wedge m_1, m_2 \neq \perp) \ ?$ **true** : **false**

By looking at the construction of $\mathcal{D}$ we notice that, $\omega_1, \omega_2 \neq \perp$ from the experiment in (12) implies $m_1, m_2 \neq \perp$. Similarly, the correctness of the commitment scheme guarantees that $\omega_1 \neq \omega_2$ implies $m_1 \neq m_2$. Thus, if P succeeds in breaking the revealing property with non-negligible chance, then $AttackCommitBinding$ returns true with the same probability, i.e., the binding property of the commitment scheme is broken.

(4) **No cheating.** Let $\Delta(\Omega)$ be the distribution of Ω induced by $\langle \alpha, \beta, \lambda \rangle$, it is not difficult to see that

$$\mathbb{E}[u_P(\omega, \omega, b_1, b_2) | (\omega, b_1, b_2) \leftarrow \textbf{Prot}(\Delta(\Omega), k); (b_1, b_2) \neq (\emptyset, \emptyset)] = -r(1+\alpha) - \gamma\lambda + \epsilon'(k) \tag{18}$$

Then, recall from information theory that if a is uniform in $\{0,1\}^x$, and b is chosen with unknown but independent distribution from $\{0,1\}^x$ then $a \oplus b$ is uniform in $\{0,1\}^x$. We notice that since the commitment scheme is secure and non-malleable, c_1 and c_2 give P negligible advantage in creating c_1' and c_2'. Thus, the construction of $\mathcal{D}$ suggests that ω_1 and ω_2 in experiment $\textbf{Real}(\Delta(\Omega), k, P)$ both have the same distribution $\Delta(\Omega)$. Moreover, we either have $\omega_1 = \omega_2$, or that they are statistically independent, i.e.,

$$\sup_{x \in \Omega} \sup_{y, y' \in \Omega} |\Pr[\omega_1 = x | \omega_2 = y] - \Pr[\omega_1 = x | \omega_2 = y']| \leq \epsilon(k)$$
$$\sup_{y \in \Omega} \sup_{x, x' \in \Omega} |\Pr[\omega_2 = y | \omega_1 = x] - \Pr[\omega_2 = y | \omega_1 = x']| \leq \epsilon(k)$$

Clearly, the former occurs when the principal is honest, i.e., it forwards the messages between agents. This gives expected utility

$$-r(1+\alpha) - \gamma\lambda + \epsilon_1(k) \tag{19}$$

where $\epsilon''(k)$ is a negligible function. The latter occurs when P modifies at least one of the commitments. Note that from the definition of u_P, the principal would not produce (b_1, b_2) such that $\emptyset \sim_1 \omega_1$ and $\emptyset \sim_2 \omega_2$, because then its utility is the worst. Thus, the situations in which P must produce $(b_1, b_2) = (\emptyset, \emptyset)$ is when $\omega_1 = (A_2, \cdot)$ and $\omega_2 = (A_1, \cdot)$. It is easy to see that these altogether occur with probability

$$\begin{aligned}
\Pr[\omega_1 = (A_2, \cdot), \omega_2 = (A_1, \cdot)] &= \Pr[\omega_1 = (A_2, \cdot)] \Pr[\omega_2 = (A_1, \cdot)] + \epsilon_2(k) \\
&= \frac{1}{2}(1-\alpha) \cdot \frac{1}{2}(1-\alpha) + \epsilon_3(k) \\
&= \frac{1}{4}(1-\alpha)^2 + \epsilon_3(k)
\end{aligned}$$

Similarly, we also have

$$\begin{aligned}
&\Pr[\omega_i = (A_{3-i}, \cdot)] = \frac{1}{2}(1-\alpha) + \epsilon_4(k) \\
&\Pr[\omega_i = (A_i, audit)] = \frac{1}{2}(1-\alpha)\frac{\lambda}{1-\alpha} = \frac{1}{2}\lambda + \epsilon_5(k) \\
&\Pr[\omega_i = (A_i, reward)] = \frac{1}{2}(1-\alpha)(1 - \frac{\lambda}{1-\alpha}) = \frac{1}{2}(1-\alpha-\lambda) + \epsilon_6(k) \\
&\Pr[\omega_i = ((A_1, A_2, reward), \cdot)] = \alpha + \epsilon_7(k)
\end{aligned}$$

The expected utility of the principal is thus

$$\begin{aligned}
&\mathbb{E}[u_P(\omega_1, \omega_2, b_1, b_2) | (\omega_1, \omega_2, b_1, b_2) \leftarrow \textbf{Real}(\Delta(\Omega), k, P); (b_1, b_2) \neq (\emptyset, \emptyset)] \\
&= \sum_{(\omega_1, \omega_2, b_1, b_2) \neq (\cdot, \cdot, \emptyset, \emptyset)} \frac{\Pr[\omega_1, \omega_2, b_1, b_2] u_P(\omega_1, \omega_2, b_1, b_2)}{1 - \Pr[\omega_1 = (A_2, \cdot), \omega_2 = (A_1, \cdot)]} \\
&= \sum_{(\omega_1, \omega_2, b_1, b_2) \neq (\cdot, \cdot, \emptyset, \emptyset)} \frac{\Pr[\omega_1] \Pr[\omega_2] u_P(\omega_1, \omega_2, b_1, b_2)}{1 - (1-\alpha)^2/4} + \epsilon_8(k) \\
&= -\frac{4(r + 3r\alpha + \gamma\lambda)}{(3-\alpha)(1+\alpha)} + \epsilon_9(k)
\end{aligned}$$

$$< -r(1+\alpha) - \gamma\lambda + \epsilon_9(k)$$

This thus concludes the no-cheating property.

□