V3rified: Revelation vs Non-Revelation Mechanisms for Decentralized Verifiable Computation

Athina Terzoglou[1], Tiantian Gong[2], Aniket Kate[1,3], and Alexandros Psomas[1]

[1] Purdue University {aterzogl,aniket,apsomas}@purdue.edu
[2] Yale University hildawithin@gmail.com
[3] Supra Research a.kate@supra.com

Abstract. In the era of Web3, decentralized technologies have emerged as the cornerstone of a new digital paradigm. Backed by a decentralized blockchain architecture, the Web3 space aims to democratize all aspects of the web. From data-sharing to learning models, outsourcing computation is an established, prevalent practice. Verifiable computation makes this practice trustworthy as clients/users can now efficiently validate the integrity of a computation. As verifiable computation gets considered for applications in the Web3 space, decentralization is an important requirement that guarantees system reliability and ensures that no single entity can suppress clients. At the same time, decentralization needs to be balanced with efficiency: clients want their computations done as quickly as possible.

Motivated by these issues, we study the trade-off between decentralization and efficiency when outsourcing computational tasks to strategic, rational solution providers. Specifically, we examine this trade-off when the client employs (1) revelation mechanisms, i.e. auctions, where solution providers bid their desired reward for completing the task by a specific deadline and then the client selects which of them will do the task and how much they will be rewarded, and (2) simple, non-revelation mechanisms, where the client commits to the set of rules she will use to map solutions at specific times to rewards and then solution providers decide whether they want to do the task or not. We completely characterize the power and limitations of revelation and non-revelation mechanisms in our model.

Keywords: Decentralization, Mechanism Design, Verifiable Computation, Web3, Computation Outsourcing

1 Introduction

Over the last twenty years, computation outsourcing has grown enormously in the Web2 space. For example, the cloud computing market size is projected to reach $980 billion in 2025 [Sta25]. Meanwhile, computation outsourcing in Web3 has also been steadily increasing since 2023, with the market of zero-knowledge proof computation outsourcing hitting $609 million in June 2024 [DA24] and verifiable on-chain randomness services processing over a million requests monthly [Swe22].

Unlike centralized cloud computing in Web2, Web3 prioritizes *decentralization*, ensuring that the system does not rely on a single party to complete a task. This reduces single points of failure of the center, such as infrequent but catastrophic crashes or compromises, selective censorship of client requests [WEY+24], determining task completion time at its will, or charging high prices. Through decentralization, no single provider can monopolize any solution platform. Aside from maintaining decentralization, considering that these computation tasks often come with time constraints (otherwise clients could perform the computations locally), *efficiency*, i.e., clients acquiring solutions as fast as possible, is a crucial property for computation outsourcing platforms. These examples motivate mechanisms that jointly incentivize broad participation (decentralization) and low latency (efficiency).

Without centralized trust, clients naturally desire to verify computation results. Verifiable computation [GGP10,PHGR16] allows a computationally limited client to delegate a computation task to more powerful and resourceful servers, while ensuring that the output can be verified efficiently. Designing mechanisms that incentivize decentralization in Web3 is still an active field. Current platforms for verifiable computation treat decentralization as given. For example, Gevulot [Net24] and Taiko [tai24] randomly select a worker for a task submitted by a client and pay the worker a posted price. Supra dVRF [Sup23] makes the *any-trust* assumption: on every set of workers, at least one of them behaves honestly. Assuming that there exist honest workers who always follow the protocol (even if they suffer monetary losses) and never crash, such mechanisms do avoid centralization. However, given the real-world monetary rewards, it is, of course, reasonable to expect that these supposedly honest workers are *rational* and take actions that maximize their utility; decentralization cannot be taken for granted.

Mechanism design, which studies how to craft a set of rules so that self-interested agents who act upon these rules arrive at a desired (for the designer) outcome, has affected a wide range of applications in Web3 space, including pricing oracle services like randomness oracles and off-chain data oracles [DDR+23,PLD23] and zero-knowledge proof markets [DA24]. The theory of mechanism design has focused almost exclusively on *revelation* mechanisms. Such mechanisms are designed so that agents' optimal action is very simple: "report your honest preferences." The designer has the complex task of finding outcomes in a way that truthfulness, as well as other guarantees (e.g., good revenue in the case of auctions), are satisfied. The focus on revelation mechanisms has been motivated by the *revelation principle,* a fundamental result in mechanism design which suggests that if some outcome is the equilibrium of a non-truthful mechanism, then one can construct a truthful mechanism to implement this outcome. However, the restriction to revelation mechanisms often comes at a loss, e.g., in non-Bayesian or prior-free settings [FH18,FHL21]. Furthermore, non-truthful or *non-revelation mechanisms*, e.g., the first-price auction, because of their simplicity, are a lot more common in practice, including the aforementioned case of computation outsourcing in Web3.

In this paper, we are interested in comparing revelation and non-revelation mechanisms. Simply put, our goal is *to characterize the trade-offs between decentralization and efficiency for both revelation and non-revelation mechanisms, in the context of outsourcing computation in Web3.*

1.1 Our contribution

Motivated by real-world computation outsourcing markets (see Related work), we introduce a simple model for studying decentralization and efficiency in the presence of strategic behavior. There is a client, with a total reward of R, who needs to outsource a verifiable computational task that must be completed before a deadline T, there are n strategic agents/solution providers. Agent i can pay a cost c_i to complete the computational task by time t_i; both c_i and t_i are known to the agent and not the client. We assume that agents with higher costs can complete tasks faster. The client can efficiently verify the correctness of the solutions. Invalid submissions are rejected and receive zero reward.

The client can employ a *revelation* or *non-revelation* mechanism to interact with and incentivize the agents to complete her task. A revelation mechanism first asks agents to report a cost and completion time (essentially, a bid), and then selects an outcome: who does the task (possibly more than one agent) and how much reward they get. A non-revelation mechanism is a set of rules for mapping solutions at specific times to rewards, e.g. "fastest solution gets the whole reward," or "three fastest solutions split the reward equally." For revelation mechanisms, we ask for truthfulness and individual rationality (honest agents have non-negative utility when participating); non-revelation mechanisms will be evaluated at their worst pure Nash equilibria.[4] Agents can take actions to maximize their utility (reward minus cost), but cannot misbehave in arbitrary ways. Namely, agent i cannot complete a task faster than time t_i.

Intuitively, an outcome is decentralized if many agents submit solutions. Of course, costs can be adversarially selected so that the reward of R does not suffice to cover the cost of even two agents. In the absence of incentives, the number of agents that submit solutions can be maximized by picking the k^* agents with the smallest cost, where k^* is the largest number such that the total cost is at most the reward R. We call k^* the *decentralization factor*, and strive to produce α-decentralized mechanism outcomes, those where the number of agents submitting solutions is at least αk^*. 1-decentralization implies that a maximum number of k^* agents submit solutions, while $\frac{1}{k^*}$-decentralization implies that (in the worst-case) only one agent submits a solution.

On the other hand, efficient outcomes are those where the first solution, among the submitted solutions, is as early as possible. And, again, one can pick costs and times adversarially so that adequately rewarding any two agents to submit solutions (i.e., $\frac{2}{k^*}$-decentralization) is arbitrarily slower than the time the fastest,

[4] Noting that pure Nash equilibria are not guaranteed to exist for arbitrary games; in our positive results we prove that the non-revelation mechanisms we propose do have pure Nash equilibria.

most expensive agent can produce. Therefore, for an accurate comparison, a better benchmark for efficiency is the best outcome for a *fixed* target decentralization. We say that an outcome is α-efficient if it is as fast as the fastest α-decentralized outcome. So, intuitively, a $\frac{1}{k^*}$-efficient outcome is one that has a solution at time $\min_{i\in[n]} t_i$, while a 1-efficient outcome is (in general) slower.

In the absence of strategic behavior, α-decentralization is compatible with α-efficiency. Since computation is efficiently verifiable, solution providers cannot behave arbitrarily; namely, they cannot submit incorrect solutions, as this is easily detectable. However, solution providers can report a higher cost, or submit a solution at a later time, if such an action increases their utility. In this paper, we explore the power and limitations of revelation and non-revelation mechanisms.

In Section 3 we study non-revelation mechanisms. We prove a tight bound of 1/2 for the optimal decentralization (Theorems 2 and 3), achieved by the simple non-revelation mechanism that equally splits the reward among all agents that submit a solution. This bound is tight even for $k^* = 2$. We prove that, as k^* grows, better results are possible. Specifically, it is possible to achieve a decentralization of $1 - \frac{1}{e} - \frac{8}{k^*}$ (which approaches $1 - 1/e \approx 0.63$), with a simple non-revelation mechanism that splits the reward among all agents that submit a solution according to a harmonic sequence, i.e., for some x, the fastest solution is rewarded $1/x$, the second fastest $\frac{1}{x+1}$, and so on. Surprisingly, this performance is also asymptotically tight! In Theorem 5 we prove that no non-revelation mechanism can have decentralization better than $1 - 1/(e^{1+\frac{e^2}{2k^*}})$, again, approaching $1 - 1/e$. Regarding efficiency, we prove that non-trivial decentralization is incompatible with any non-trivial efficiency guarantee for non-revelation mechanisms (Theorem 6). However, this impossibility can be circumvented by imposing additional structure on the agents' types, such that an agent's true computation time implies her true cost of computation. In this case, decentralization and efficiency can be combined optimally, as if agents were non-strategic (Theorem 7).

In Section 4 we study revelation mechanisms. We first show that, with respect to decentralization, revelation mechanisms face the same barriers as non-revelation mechanisms: in an analogue to Theorem 2, in Theorem 8 we prove that there is no incentive compatible and individually rational, α-decentralized revelation mechanism, for any constant $\alpha > 1/2$. However, efficiency and decentralization are compatible for revelation mechanisms. We give a mechanism parameterized by k that admits a truthful ex-post Nash equilibrium and achieves $\frac{\min\{k,k^*\}-1}{k^*}$-decentralization and $\frac{\min\{k,k^*\}}{k^*}$-efficiency (Theorem 9). Our mechanism, Inverse Generalized Second Price (I-GSP), resembles the generalized second price auction (GSP), a well-studied *non-truthful mechanism* in auction theory [EOS07]. In GSP, the i-th highest bidder is charged the $(i+1)$-th highest bid. In I-GSP, the agent with the i-th smallest cost is rewarded the $(i+1)$-th smallest cost, with a twist to include a fast (and expensive) bidder, in order to guarantee good efficiency.

In Section 5, we experimentally evaluate our algorithms through two sets of experiments. In the first set, we compare the decentralization factor of the equal reward mechanism R^{eq} (Section 3.1) and the harmonic mechanism R^{harm} (Section 3.2). Our results show that both mechanisms consistently outperform

their theoretical worst-case guarantees on these random inputs. In the second set, we compare the fastest solution under the non-revelation harmonic mechanism R^{harm} with the revelation mechanism inverse generalized second price $\mathcal{M}^{\text{I-GSP}}$ (Section 4), using the same number of participants. The results confirm that $\mathcal{M}^{\text{I-GSP}}$, that prioritizes efficiency, consistently outperforms R^{harm}.

At a high level, our results highlight the differences between revelation and non-revelation mechanisms. On the one hand, time-based non-revelation rules are simple to implement, since agents either submit a valid solution (or not), and the mechanism derives all rewards from the set of submissions and their order. There is no separate bidding or reporting phase. However, in this simple model, decentralization and efficiency are incompatible. Furthermore, in a non-equilibrium outcome, an agent may have negative utility. In contrast, revelation mechanisms introduce a reporting phase in which agents submit bids (i.e., declared costs and completion times), and the mechanism computes an allocation and payments based on these bids. This additional communication overhead comes with two benefits. First, the designer can achieve both decentralization and efficiency. Second, the designer can ensure that every agent who is allocated the task is paid at least her cost.

1.2 Related work

Fee mechanism for computation outsourcing markets in Web3. In the proof market of =nil; [nF24], a client can submit a transaction specifying the time and price for the request, and whether she desires to prioritize time or price. Solution providers then submit bids, and are ranked according to their proof generation time and price (in the order specified by the client). The top provider is the selected solver. Taiko [tai24] assigns weights to providers, which are positively correlated with providers' stakes and negatively correlated with their asking prices. For a request, it selects a provider at random according to their weights. Wang et al. [WZY+25] design a mechanism for prover markets for achieving efficiency (social welfare), incentive compatibility, and collusion resistance. The clients are selected via a "pay-as-bid greedy auction," i.e., picking the client transactions that maximize the payment amounts while conforming to a finite batch capacity constraint. The providers are then selected as solvers uniformly at random. All the above approaches treat decentralization as given. Succinct network [suc24] utilizes an all pay auction [BKDV96] to allocate proof tasks and rewards: Given m providers, each provider i that bids an amount b_i pays their bid and wins the auction – thus the prize from the client – with probability $\frac{b_i^\alpha}{\sum_{j=1}^m b_j^\alpha}$ where α is a system parameter. Chaidos et al. [CKM23] propose a game theoretic model of blockchain participation games for block production and related blockchain tasks.

Centralization issues in Web3. Heimbach et al. [HKFTW23] point out the concentration of Ethereum's relay and builder industry by measuring the Herfindahl-Hirschman Index (HHI) in the two industries. More straightforward statistics also exhibit a centralization trend: The top three MEV-boost [Fla24b]

builders in Ethereum can produce more than 90% of the blocks[Fla24a]; The top five mining pools in Bitcoin can produce approximately 80% of the blocks [Blo24]. Previous works [GKCC13,GBE+18,OKK24] have also more systematically highlighted the concentration of hardware manufacturers, codebases, mining powers, R&D funding, decision-making, and incident resolution processes in blockchains. Wahrstaetter et al. [WEY+24] study censorship by a single entity in prominent permissionless blockchains, and observe that censorship by centralized entities affects not only neutrality but also the security of blockchains.

Revelation and non-revelation mechanism design. While revelation mechanisms are the dominating paradigm in mechanism design, recent results show that non-revelation mechanisms have surprising theoretical advantages in canonical domains, e.g., prior-independent mechanism design [FH18,FHL21]. Non-revelation mechanisms are prevalent in practice, e.g., the first price auction. A thematically related (to our paper) line of work that studies the design of *contests*, where participants incur a cost and produce an output of a certain quality also, almost exclusively, focuses on non-revelation mechanisms and their equilibria, e.g., [MS01]; see [Cor07] for a survey. A major difference with our paper is that contest design ignores decentralization and primarily focuses on maximizing the expected maximum quality or sum of utilities.

2 Preliminaries

There is a client interested in outsourcing a computational task that needs to be finished by time T, and is willing to provide a reward of R; without loss of generality, we normalize this reward to $R = 1$. There are n solution providers that can compute solutions to the client's task; throughout the paper, we refer to them as agents. Agent i has a cost c_i that she can incur to compute a solution; after she pays this cost, it takes her $t_i \leq T$ time to compute the solution. We refer to the tuple (c_i, t_i) as agent i's *type*. Agents' types are private. We assume that agents with higher costs can complete tasks faster, i.e., if $c_i > c_j$ then $t_i < t_j$. This assumption is only used in our positive revelation-mechanism result; all other results go through even without this assumption. To ensure feasibility, we further assume there is at least one agent with $c_i < 1$, so that the reward can indeed cover her cost. Agents have quasi-linear utilities; that is, if agent i gets reward r_i and pays a cost c_i, her utility is $r_i - c_i$. Agents are rational and will take actions that maximize their utility. For example, in both revelation and non-revelation mechanisms, agents always have the option of not participating, which we will think of as taking the action $\perp$, if their (expected) reward is strictly less than their cost. Submitted solutions are efficiently verifiable. Invalid submissions are discarded and yield zero reward, while the agent still incurs the computation cost. Our analysis, therefore, focuses on participation and completion times, rather than the correctness of the solutions submitted.

Revelation and non-revelation mechanisms. The client can outsource her task by deploying a *revelation*, or a *non-revelation* mechanism.

For non-revelation mechanisms, we consider mechanisms induced by a function that maps a vector of solutions at specific times to rewards. We denote with $\perp$ the action of not submitting a solution. A rule $\mathcal{R}(a_1, \ldots, a_n)$ outputs a reward vector $(r_1, \ldots, r_n)$, where a_i is the time agent i submitted her solution, or $\perp$ if i did not submit a solution within time T. We write $\mathcal{R}_i(a_1, \ldots, a_n)$ for the reward of agent i under rules $\mathcal{R}$, when the actions taken are $a_1, \ldots, a_n$. Agent i cannot compute solutions in time less than t_i. Therefore, we have that for all $i \in [n]$, $a_i \geq t_i$ (or $a_i = \perp$). Depending on $\mathcal{R}$, agent i might maximize her utility by submitting a solution at a time later than t_i, or by not submitting a solution at all. For non-revelation mechanisms, an outcome is a Nash equilibrium. Throughout, we analyze the induced game under complete information, meaning that agents know the type profile $(c_i, t_i)_{i\in[n]}$ when choosing whether and when to submit, while the rule $\mathcal{R}$ itself does not observe types. We will evaluate non-revelation mechanisms via their (worse) *pure* Nash equilibria. For agents with types $(c_1, t_1), \ldots, (c_n, t_n)$, the strategy profile $(a_1, \ldots, a_n)$ is a pure Nash equilibrium under $\mathcal{R}$, if no agent wants to deviate to a different (feasible) action a_i', i.e. for every agent $i \in [n]$, $\mathcal{R}_i(a_i; a_{-i}) - c_i \cdot 1\{a_i \neq \perp\} \geq \mathcal{R}_i(a_i'; a_{-i}) - c_i \cdot 1\{a_i' \neq \perp\}$, where $1\{\cdot\}$ is the indicator function and a_{-i} is the set of actions taken by all agents except i. We will assume that, if an agent is indifferent between submitting and not submitting a solution, then they prefer submitting. Since we consider (worst-case) pure Nash equilibria, which don't always exist, in our positive results for non-revelation mechanisms, we also show that pure Nash equilibria exist.

Example 1 (Reward the fastest solution). Let $\mathcal{R}^{\text{Fast}}$ be the non-revelation mechanism that gives the whole reward to the agent that submitted a solution first, breaking ties lexicographically. Then, $\mathcal{R}_i(a_1, \ldots, a_n)$ is 1 only for the fastest agent, that is $i = \arg\min_{j\in[n]:a_j\neq\perp}\{a_j\}$, and 0 otherwise. A pure Nash equilibrium of $\mathcal{R}^{\text{Fast}}$ is the (lexicographically in case of a tie) fastest agent, $i^* \in \arg\min_{i\in[n]} t_i$, to take action $a_{i^*} = t_i$, and $a_i = \perp$ for all $i \neq i^*$.

A revelation mechanism $\mathcal{M} = (x, r)$ consists of two functions: an allocation function x and a reward function r, which map *reported* types, or bids, to allocations and rewards. Specifically, both functions take as input agents' bids: the bid $b_i = (\hat{c}_i, \hat{t}_i)$ of agent i consists of a cost $\hat{c}_i$ (possibly different to c_i) and a time $\hat{t}_i$ (possibly different to t_i). Let $\mathbf{b} = (b_1, \ldots, b_n)$ denote a vector of bids. $x_i(\mathbf{b})$ denotes the probability that agent i is selected for the task and $r_i(\mathbf{b})$ is the reward of agent i under bids $\mathbf{b}$. We consider the allocation $x(\mathbf{b})$ and the reward $r_i(\mathbf{b})$ to be the outcome of the revelation mechanism $\mathcal{M}$. Similar to the case of non-revelation mechanisms, we assume that agent i cannot report a time smaller than t_i, i.e., $\hat{t}_i \geq t_i$, since solving faster than her true time is infeasible, yielding zero reward. A mechanism can select multiple agents to perform the task, i.e., $\sum_{i\in[n]} x_i(\mathbf{b})$ could be (and decentralization implies that it is) larger than 1. Since the total available reward is 1, it must be that $\sum_{i\in[n]} r_i(\mathbf{b}) \leq 1$.

A revelation mechanism is *incentive compatible*(henceforth, **IC**) if no agent has an incentive to misreport their type, assuming all other agents report their type honestly. That is, a revelation mechanism is IC if truthtelling is a Nash

equilibrium i.e., for all types $\mathbf{b} = ((c_1, t_1), \ldots, (c_n, t_n))$ and for all b_i it holds that $r_i((c_i, t_i); \mathbf{b}_{-i}) - c_i \cdot x_i((c_i, t_i); \mathbf{b}_{-i}) \geq r_i(b_i; \mathbf{b}_{-i}) - c_i \cdot x_i(b_i; \mathbf{b}_{-i})$. A mechanism is *individually rational* (henceforth, **IR**) if every agent gets non-negative utility when reporting her true type, i.e., for all $i \in [n]$, for all types (c_i, t_i), and for all $\mathbf{b}_{-i}$ it holds that $r_i((c_i, t_i); \mathbf{b}_{-i}) - c_i \cdot x_i((c_i, t_i); \mathbf{b}_{-i}) \geq 0$. We assume throughout the paper that revelation mechanisms ignore agents that are "dominated," in the sense of having higher cost and slower time than other agents (since those bids can only arise from dishonest behavior).

Example 2 (Inverse k-price auction). Assume that the reported costs satisfy $c_1 \leq \cdots \leq c_n$, let $\mathcal{M}^{\text{I-}k \text{ price}}$ be the revelation mechanism that allocates the task to the k agents with the smallest (reported) costs, where k is the largest integer such that $k\, c_{k+1} \leq 1$. The reward of agent i is c_{k+1}, for all $i = 1, \ldots, k$ (i.e., the mechanism is an inverse k price auction). $\mathcal{M}^{\text{I-}k \text{ price}}$ is individually rational ($c_i \leq c_{k+1}$) and IC, since if agent i increases her reported cost her reward either stays the same or she is ignored (due to observable dishonest behavior: having a high cost and slow time), and if agent i decreases her reported cost her reward is at most c_{k+1} (or c_k for $i > k$, in which case this reward is not sufficient).

Decentralization and Efficiency. The *decentralization factor* is the maximum number of agents that can be given a reward that covers their cost.

Definition 1 (Decentralization factor k^*). *For a set of n agents with types $(c_1, t_1), \ldots, (c_n, t_n)$, the decentralization factor k^* is the size of the largest set of agents that can be given a reward at least as large as their cost. Formally, $k^* = \max\{|S| : S \subseteq [n], \sum_{i \in S} c_i \leq 1\}$.*

We use k^* as a benchmark for decentralization since it is precisely the maximum number of agents that the system could adequately reward, in the absence of strategic behavior. The decentralization factor is a characteristic of the instance, and, hence, independent of any mechanism deployed. As we see in the technical sections, perfect decentralization is not compatible with strategic behavior, and therefore, we seek approximately decentralized outcomes.

Definition 2 (α-decentralization). *The outcome of a mechanism is α-decentralized, for $\alpha \in (0, 1]$, if at least αk^* agents are submitting solutions; a mechanism (revelation or non-revelation) is α-decentralized, $\alpha \in (0, 1]$, if all its outcomes are α-decentralized.*

We are interested in balancing decentralization with efficiency (fast solutions). In terms of efficiency, a first benchmark could be the fastest time an agent can submit a solution, i.e. $\min_{i \in [n]} t_i$. However, this benchmark might not even be approximately possible when combined with nontrivial decentralization, since the fastest agent might require the entire reward for herself. Therefore, we define efficiency in terms of the fastest solution of an α-decentralized outcome.

Definition 3 (Time guarantee t^*_α). *Given types $(c_1, t_1), (c_2, t_2), \ldots, (c_n, t_n)$ of n agents, the time guarantee t^*_α, for $\alpha \in (0, 1]$, is the fastest α-decentralized set where agents are rewarded at least their cost, $t^*_\alpha = \min_{S : |S| \geq \alpha k^*, \sum_{i \in S} c_i \leq 1} \min_{i \in S} t_i$.*

For example, consider an instance with agent types $(1, \epsilon)$, $(1/2, 1)$ and $(1/2, 1)$. Here, $k^* = 2$. t_1^* is the fastest time in any set with $k^* = 2$ participating agents; in the example, $t_1^* = 1$. Similarly, $t_{1/2}^*$ is the fastest time in any set with $\frac{1}{2}k^* = 1$ participating agents; in the example, $t_{1/2}^* = \epsilon$. In general, as the following proposition shows, as α becomes smaller, t_α^* becomes smaller as well.

Proposition 1. *t_α^* is a monotone non-decreasing function of α. That is, for all $\alpha < \alpha'$, $t_\alpha^* \leq t_{\alpha'}^*$.*

Definition 4 (β-efficiency). *We say that an outcome with a set S of participating agents is β-efficient if $\min_{i \in S} t_i \leq t_\beta^*$; a mechanism (revelation or non-revelation) is β-efficient, $\beta \in (0, 1]$, if all its outcomes are β-efficient.*

Note that, perhaps counterintuitively, as β becomes smaller, β-efficiency is a *stronger* guarantee. That is, an ϵ-efficient mechanism, for a small $\epsilon > 0$, has faster solutions than a 1-efficient mechanism. The following definition and claim will be useful when arguing the efficiency of our mechanisms.

Definition 5 (k-Best Set). *For agents with types $(c_1, t_1), \ldots, (c_n, t_n)$, such that $c_1 \leq \ldots \leq c_n$, and for a $k \leq k^*$ the* k-best set *is defined as*

$$S_k^* = \{1, \ldots, k-1\} \cup \left\{ \underset{i \notin \{1,\ldots,k-1\}}{\arg\min} \left(t_i : c_i + \sum_{j=1}^{k-1} c_j \leq 1 \right) \right\}$$

The k-Best set is not unique, but all k-Best sets have the same total cost. By construction the k-Best set is also $\frac{k}{k^*}$-efficient. The proof is in the Appendix A.

Claim 1. *For agents with types $(c_1, t_1), \ldots, (c_n, t_n)$, such that $c_1 \leq \ldots \leq c_n$, with decentralization factor k^*, for any $k \leq k^*$ the k-Best set is $\frac{k}{k^*}$-efficient.*

In the absence of strategic behavior, it is possible to simultaneously satisfy α-decentralization and α-efficiency. Our definitions allow the comparison of an α-decentralized outcome (αk^* agents participating) with a β-efficient outcome (the fastest set with βk^* participating agents). For $\beta > \alpha$ the monotonicity of the time guarantee, Proposition 1, implies $t_\beta^* \geq t_\alpha^*$.

3 Non-Revelation Mechanisms

In this section, we study non-revelation mechanisms. We first prove a simple, tight bound of $1/2$ for the optimal decentralization (Theorems 2 and 3). This bound can be improved to $1 - 1/e$ (also in a tight way, under a natural assumption) as k^* grows (Theorems 4 and 5). In Section 3.3 we prove that, unfortunately, non-trivial decentralization is incompatible with efficiency (Theorem 6). Finally, we show how to bypass Theorem 6 in a very strong way: assuming some structure on the agents' types, decentralization and efficiency can be combined optimally, as well as in the case where agents are non-strategic (Theorem 7).

3.1 Warm-up: the Equal Reward mechanism is asymptotically optimal

We start by proving a simple, *tight* bound for decentralization: in the worst-case, one cannot incentivize (in equilibrium) strictly more than $k^*/2$ agents to participate in a non-revelation mechanism (Theorem 2). A simple non-revelation mechanism, rewarding all participating agents equally, has, in equilibrium, at least $k^*/2$ participants (Theorem 3).

Theorem 2. *There is no α-decentralized non-revelation mechanism, for any constant $\alpha > \frac{1}{2}$.*

Proof. Suppose there is a non-revelation mechanism $\mathcal{R}$ that is α-decentralized, for some $\alpha > \frac{1}{2}$. Since α-decentralization requires the guarantee to hold for every instance, it suffices to show a contradiction for $n = 2$ (with $k^* = 2$). Consider the the following instance with $n = 2$ agents whose types are $(1/2, T)$ and $(1/2, T)$. Since $k^* = 2$, and since $\mathcal{R}$ is α-decentralized for $\alpha > \frac{1}{2}$, it must be that in all equilibria of $\mathcal{R}$, both agents submit solutions. To cover their cost, both must be rewarded at least $1/2$ (and therefore exactly $1/2$, since the total reward is at most 1). Second, since the agents cannot submit a solution after time T, or before T (since their type is $(., T)$), the equilibrium is such that they both submit a solution at time T. Therefore, we can conclude that $\mathcal{R}(T, T) = (1/2, 1/2)$.

Now, consider the case of $n = 2$ agents whose types are $(2/3, T)$ and $(1/3, T)$. Again, $k^* = 2$, and since $\mathcal{R}$ is α-decentralized for $\alpha > \frac{1}{2}$, it must again be that in all equilibria of $\mathcal{R}$, both agents submit solutions. Agents can only submit a solution at time T (or not at all), and since $\mathcal{R}(T, T) = (1/2, 1/2)$, the first agent, whose cost is $2/3 > 1/2$, prefers to not submit a solution if the other agent submits one. Therefore, $\mathcal{R}$ can only have one agent submitting a solution in equilibrium; a contradiction. □

Theorem 3 (Equal Reward Rule). *Let $\mathcal{R}^{eq}$ be the non-revelation mechanism that rewards all participating agents equally. That is, $\mathcal{R}_i^{eq}(a_1, \ldots, a_n) = 1/\ell$ for all i with $a_i \neq \perp$, where $\ell = |\{i \in [n] : a_i \neq \perp\}|$. Then, $\mathcal{R}^{eq}$ is $\frac{1}{2}$-decentralized.*

Proof. First, we prove that all pure Nash equilibria of $\mathcal{R}^{\text{eq}}$ have the same number of agents. Then, given an arbitrary instance, we construct a pure Nash equilibrium[5] of size at least $\frac{k^*}{2}$. Combined, the two statements prove the theorem.

Towards the first statement, assume that $S_A \subseteq [n]$ and $S_B \subseteq [n]$ are the agents that submit solutions in two equilibria of $\mathcal{R}^{\text{eq}}$, such that $|S_A| + 1 \leq |S_B|$. By definition, $\forall i \in S_A\ c_i \leq 1/|S_A|$ and $\forall j \in S_B\ c_j \leq 1/|S_B|$. There exists an agent i^* such that $i^* \in S_B \backslash S_A$, and since $i^* \in S_B$ we have that $c_{i^*} \leq 1/|S_B| \leq 1/(|S_A|+1)$. Therefore, only the set S_A submitting a solution cannot be an equilibrium, since i^* prefers submitting a solution to not submitting a solution;[6] a contradiction.

[5] Thus, also showing that pure Nash equilibria exist for $\mathcal{R}^{\text{eq}}$.

[6] Recall agents prefer submitting a solution over not submitting when their expected utilities are equal.

Towards the second statement, we construct an equilibrium as follows. First, without loss of generality, rename the agents such that $c_1 \leq \ldots \leq c_n$. Notice that k^* is the largest index k such that $\sum_{i=1}^{k} c_i \leq 1$. Let ℓ be the largest index such that $c_\ell \leq \frac{1}{\ell}$ and $c_{\ell+1} > \frac{1}{\ell+1}$ (and $\ell = n$ if $\forall i, c_i \leq 1/i$). By construction, the set of agents $\{1, 2, \ldots, \ell\}$ submitting a solution is an equilibrium of $\mathcal{R}^{\text{eq}}$. Finally, $1 \geq \sum_{i=1}^{k^*} c_i = \sum_{i=1}^{\ell} c_i + \sum_{i=\ell+1}^{k^*} c_i \geq \sum_{i=\ell+1}^{k^*} c_i > \frac{k^*-\ell}{\ell+1}$. Re-arranging, we have $2\ell > k^* - 1$. Since ℓ and k^* are integers we get $2\ell \geq k^*$. Hence, we have $\ell \geq \frac{k^*}{2}$. □

3.2 The Harmonic Rule: better decentralization for large k^*

Theorems 2 and 3 give a tight bound on the decentralization guarantees of non-revelation mechanisms. However, instances where this bound is witnessed involve small values of k^*, specifically $k^* = 2$. We prove that, when k^* is large, better guarantees are possible. The harmonic rule is inspired by a simple observation about the costs of the k^* cheapest agents (w.l.o.g., assume $c_1 \leq c_2 \leq \ldots \leq c_{k^*}$). Since the total reward is normalized to 1, the sum of these costs is at most 1, which implies that they must satisfy bounds of the form $c_1 \leq 1/k^*$, $c_2 \leq 1/(k^*-1)$, and so on. The harmonic rule then allocates rewards to the fastest participating agents according to a decreasing sequence of the form $1/m, 1/(m+1), 1/(m+2), \ldots$, for some offset m chosen so that the total reward is at most 1. In this way, the i-th cheapest agent can be rewarded at least her cost, and is therefore incentivized to participate. As k^* grows, this harmonic structure allows the mechanism to keep a large fraction of the k^* cheapest agents in equilibrium across very different cost profiles, which underlies the improved decentralization guarantee proved below. The proofs are deferred to Appendix A.

Theorem 4 (Harmonic Rule). *Let $\mathcal{R}^{Harm}$ be the non-revelation mechanism that rewards the i-th fastest participating agent $\frac{1}{m+i-1}$, where m is the smallest non-negative integer for which the overall reward is at most 1. That is, if $\ell = |\{i \in [n] : a_i \neq \perp\}|$ is the number of agents participating, and agent j submitted the i-th fastest solution, the reward of j is $\frac{1}{m+i-1}$, where $m = \arg\min_{a \in \mathbb{N}} \{\sum_{x=a}^{a+\ell-1} \frac{1}{x} \leq 1\}$. Then, $\mathcal{R}^{Harm}$ is $\left(1 - \frac{1}{e} - \frac{8}{k^*}\right)$-decentralized.*

The guarantee in Theorem 4 is also (asymptotically) tight, under the following condition. We say that a non-revelation mechanism is **reward-monotone** if the i-th smallest reward when ℓ agents submit solutions is at least the i-th smallest reward when $\ell + 1$ agents submit solutions.

Theorem 5. *There is no α-decentralized and reward-monotone non-revelation mechanism, for any $\alpha > 1 - \left(e^{1+\frac{e^2}{2k^*}}\right)^{-1}$.*

3.3 Decentralization, efficiency, and non-overlapping types

So far, we have focused on decentralization and ignored efficiency. Optimal efficiency is trivial, by implementing the non-revelation mechanism that only

rewards the fastest solution; however, this achieves trivial decentralization of $1/k^*$. Our next result shows that non-trivial decentralization is incompatible with efficiency. Recall that 1-efficiency is the *weakest* efficiency guarantee: it asks that one of the submitted solutions (in equilibrium) is as fast as the fastest solution in the best set with k^* submitted solutions (i.e. a set with optimal decentralization).

Theorem 6. *There is no α-decentralized and 1-efficient non-revelation mechanism, for any $\alpha > \frac{1}{k^*}$.*

Proof. Suppose that there is a non-revelation mechanism $\mathcal{R}$ that is simultaneously α-decentralized and 1-efficient, for some $\alpha > \frac{1}{k^*}$.

Consider the rewards $\mathcal{R}$ produces when there are $\ell \geq 1$ solutions at time T and one solution at time $t < T$; slightly overloading notation, let $\mathcal{R}_1(\ell, t)$ be the reward of the agent that submits the solution at time t, and let r_{max} be the maximum reward this agent can receive in an equilibrium where $\ell \geq 1$ other agents submit solutions at time T. If such an equilibrium does not exist, then α-decentralization is violated, since we could pick an instance with k^* large enough so that $\alpha k^* > 1$.

Since r_{max} is the reward of one agent in an equilibrium with multiple participating agents, it must be that $r_{max} < 1$. Let $r_{max} = 1 - \epsilon^*$. Next, consider the case where there are $k^* - 1$ agents with type (ϵ, T) and one agent with type $(1 - (k^* - 1)\epsilon, \delta)$, for a small $\delta > 0$. 1-efficiency implies that the fast agent must be participating in all equilibria. We can pick k^* large enough so that α-decentralization will imply that all equilibria have at least two participating agents. However, by picking ϵ such that $1 - \epsilon^* < 1 - (k^* - 1)\epsilon$, we have that r_{max}, the maximum possible reward in an equilibrium, is not an adequate reward for the fast agent. Therefore, no equilibrium where this agent and at least one more agent participate exists; a contradiction. □

Theorem 6 shows that decentralization and efficiency are not compatible when agents are strategic. Intuitively, what goes wrong is that the fastest agent *must* be incentivized to participate, but there is no way to reward her adequately while also rewarding a second agent. To bypass this negative result, here we assume that agents' types satisfy the following structure on types, which we call **non-overlapping types**. Consider m profile "buckets," $B = \{([\underline{t}_1, \bar{t}_1], \hat{c}_1), \ldots, ([\underline{t}_m, \bar{t}_m], \hat{c}_m)\}$, characterized by a computation time interval and an upper bound on computation costs. The costs are distinct and monotone: $\hat{c}_1, < \ldots < \hat{c}_m$. Furthermore, there are no overlapping time intervals, that is, for all $j < m$, $\bar{t}_{j+1} < \underline{t}_j$. Overloading notation, we say $t \in B_j$ if $t \in [\underline{t}_j, \bar{t}_j]$. For an agent with computation time t, let $b(t) = \{j : t \in B_j\}$ $(b(\perp) = \perp)$ be the function that returns her bucket and $c(t) = \hat{c}_{b(t)}$ $(c(\perp) = \perp$ be the function that returns her cost. In this model, a non-revelation mechanism can leverage the fact that the time of submission is correlated with the cost in order to achieve decentralization and efficiency simultaneously.

Our non-revelation mechanism, $\mathcal{R}^{\text{Best-Set}}$, takes as input a parameter $k \leq k^*$ (the target size of the outcome) and works as follows. If the number of solutions is $m' < k$, $\mathcal{R}^{\text{Best-Set}}$ rewards the z cheapest agents exactly their cost (for z as large

as possible). Otherwise, $\mathcal{R}^{\text{Best-Set}}$ finds the k-best set of $(c(a_1), a_1), \ldots, (c(a_n), a_n)$ with the fastest $k-1$ agents with the smallest cost. Formally, to find the k-best set, first, $\mathcal{R}^{\text{Best-Set}}$ partitions the agents into their corresponding buckets $\hat{B}_j = \{i : b(a_i) = j\}$ ($\hat{B}_\perp = \{i : b(a_i) = \perp\}$). Then, it picks the $k-1$ (fastest) agents with the smallest cost. For $\hat{B}_1, \ldots, \hat{B}_m, \hat{B}_\perp$, let $z^* = \arg\min_z\{\sum_{j=1}^{z} |\hat{B}_j| > k-1\}$ be the index of the first bucket that cannot be rewarded in its entirety, and let $S^{(1)} = \cup_{j=1}^{z^*-1} \hat{B}_j$ and $S^{(2)} \subset \hat{B}_{z^*}$ be the fastest agents such that $|S^{(1)} \cup S^{(2)}| = k-1$. Finally, let $\ell = \arg\min_i\{t_i : \sum_{j \in S^{(1)} \cup S^{(2)}} c_j \leq 1\}$. The reward of agent i is $c(a_i) \cdot \mathbb{1}\{i \in S_k^*\}$, where $S_k^* = S^{(1)} \cup S^{(2)} \cup \{\ell\}$. The proof is deferred to Appendix A.

Theorem 7 (Best-Set Rule). *When types are non-overlapping, $\mathcal{R}^{\text{Best-Set}}$, on input $k \leq k^*$, is $\frac{k}{k^*}$-decentralized and $\frac{k}{k^*}$-efficient.*

4 Revelation Mechanisms

In this section, we study revelation mechanisms. We first prove an analogue to Theorem 2, where we show that revelation and non-revelation mechanisms have the same decentralization lower bound. The proof is in Appendix A.

Theorem 8. *No IC and IR revelation mechanism is α-decentralized, for any constant $\alpha > 1/2$.*

The fact that the inverse k-price auction of Example 2 is, in fact, $1/2$ decentralized (see Appendix A for a proof) implies that Theorem 8 is tight. Similar to Theorem 2, the bound of $1/2$ can be improved. However, unlike non-revelation mechanisms, optimal efficiency is compatible with optimal decentralization.

Our next result shows a *revelation gap*: Theorem 6 is not true for revelation mechanisms. We give a mechanism parameterized by k that is incentive compatible (IC) and achieves $\frac{\min\{k,k^*\}-1}{k^*}$-decentralization and $\frac{\min\{k,k^*\}}{k^*}$-efficiency. Our mechanism, Inverse Generalized Second Price (I-GSP), denoted by $\mathcal{M}^{\text{I-GSP}}$, takes as input a parameter k and works as follows. To simplify the description of the mechanism, assume for now that $k \leq k^*$. Without loss of generality, reported costs satisfy $c_1 \leq \cdots \leq c_n$. If there is an inversion, i.e., there is an $i > j$ such that $t_i < t_j$, the mechanism does not allocate anything. Otherwise, $\mathcal{M}^{\text{I-GSP}}$ allocates the task to the $k-2$ agents with the smallest cost, as well as the agent ℓ with the highest cost possible to maintain feasibility, i.e., $\ell = \arg\max_z c_z + \sum_{i=2}^{k-1} c_i \leq 1$. Agent ℓ gets reward $1 - \sum_{i=2}^{k-1} c_i$; for $i = 1, \ldots, k-2$, the reward of agent i is c_{i+1}. If $k > k^*$, then $\sum_{i=1}^{k} c_i > 1$, so the mechanism follows the procedure above for k^* instead of k, where $k^* = \arg\max_\ell\{\sum_{i=1}^{\ell} c_i \leq 1\}$ of the reported costs.

Theorem 9 (Inverse Generalized Second Price (I-GSP)). *$\mathcal{M}^{\text{I-GSP}}$ is IC, IR, and, on input k, is $\frac{\min\{k,k^*\}-1}{k^*}$-decentralized and $\frac{\min\{k,k^*\}}{k^*}$-efficient revelation mechanism.*

Proof. First, we prove that $\mathcal{M}^{\text{I-GSP}}$ is individually rational. It is easy to see that for agents $i = 1, \ldots, k-2$ the reward $r_i = c_{i+1} \geq c_i$. Therefore, it suffices to show that $c_\ell \leq r_\ell = 1 - \sum_{i=2}^{k-1} c_i$, which is true by construction since $c_\ell + \sum_{i=2}^{k-1} c_i \leq 1$.

Second, we prove that $\mathcal{M}^{\text{I-GSP}}$ is IC. Consider agent i, who reports $(\hat{c}_i, \hat{t}_i)$. Recall that no agent can under-report her time ($\hat{t}_i < t_i$) without detection, since she cannot deliver the solution faster. Delaying her solution beyond t_{i-1}, with $\hat{c}_i = c_i$, will cause a detectable inversion yielding zero utility. Regarding cost, for agents $i = 1, \ldots, k-2$, any report with $\hat{c}_i > c_{i+1}$ either creates an inversion ($\hat{t}_i = t_i$), or if paired with $\hat{t}_i < t_{i+1} < t_i$ is detected; in either case the utility is zero. Any $\hat{c}_i \in [c_i, c_{i+1})$ does not change her allocation or reward. Decreasing her reported cost can only reduce her reward if it does not cause an inversion. If $\ell > k-1$, for agents $i = k-1, \ldots, \ell-1$ reducing her reported cost below c_{k-2} can only result in a reward of at most $c_{k-2} \leq c_i$; increasing her cost might change the choice of ℓ but will not give her any reward, since $\sum_{i=2}^{k-1} c_i + c_\ell \leq 1$ (and agents cannot increase their cost above c_ℓ, without causing an inversion). Agent ℓ cannot improve her reward, since she receives the residual reward. Finally, agents $i \geq \ell + 1$ can get a reward of $1 - \sum_{i=2}^{k-1} c_i < c_{\ell+1}$ by lowering their cost.

Third, $\sum_{i=1}^{n} x_i(t_1, \ldots, t_n) = k-1$, or $k^* - 1$ if $k > k^*$, therefore $\mathcal{M}^{\text{I-GSP}}$ is $\frac{\min\{k,k^*\}-1}{k^*}$-decentralized. Fourth, by definition, agent ℓ is the fastest agent, among all sets with $\min\{k, k^*\}$ participants; hence, $\mathcal{M}^{\text{I-GSP}}$ is $\frac{\min\{k,k^*\}}{k^*}$-efficient. □

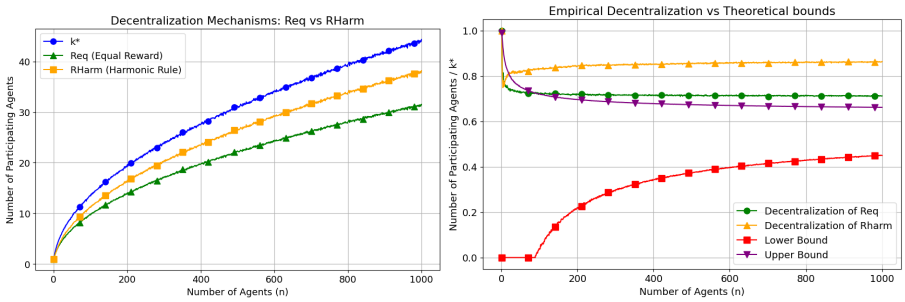
5 Experiments

In this section, we evaluate the performance of our mechanisms using synthetic data in two sets of experiments. In the first set, we compare the decentralization factor (i.e., the maximum number of participants that can be adequately rewarded) with the number of participants under the equal reward mechanism, R^{eq}, and the harmonic mechanism, R^{harm}. In the second set, we compare the efficiency (i.e., the fastest solution) of the non-revelation harmonic mechanism, R^{harm}, and the inverse generalized second price revelation mechanism, $\mathcal{M}^{\text{I-GSP}}$.

In the first set of experiments, we sample the agents' costs from two different probability distributions: Uniform $(0, 1)$ and Exponential with the parameter $\lambda = 1$ (and normalized to $(0, 1)$). These distributions capture different scenarios about the type of participating agents. In the Uniform case, all agents are assumed to be identical. The Exponential distribution represents the scenario when the majority of the agents have low costs (e.g., normal computers) and only a few agents have high costs (e.g., supercomputers).

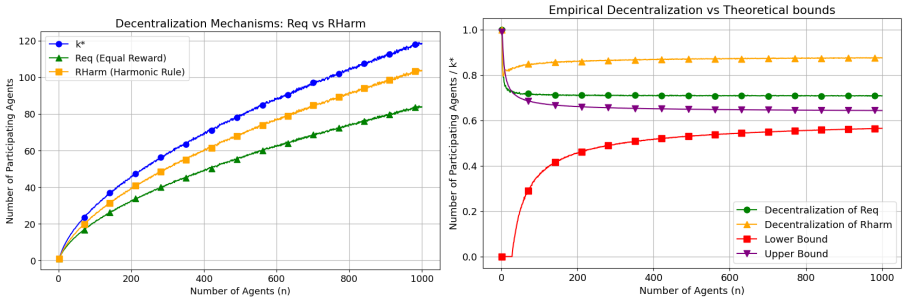
We perform simulations for instances with n participating agents, where $n \in [1, 1000]$. For each instance, we compute the decentralization factor k^*, which represents the maximal set of participating agents that can be adequately rewarded. We then compute the number of agents participating in equilibrium under each mechanism, and the corresponding empirical decentralization, which is the ratio of participating agents to k^*. The experiments are repeated 500 times for each n, and the average values are shown. In Figure 1a and Figure 2a show the number of participants in equilibrium for each mechanism (orange square for R^{harm} and green triangle for R^{Eq}) alongside k^* (blue circles). In Figures 1b and 2b, we plot the empirical decentralization of R^{harm} (orange triangles) and

R^{Eq} (green circles). We compare these empirical decentralization levels with the theoretical worst-case upper bounds (purple inverted triangles) and lower bounds (red squares). Since these bounds are proved for adversarially chosen type profiles, whereas our experiments average over random instances drawn from the well-behaved distributions described above, it is natural that the empirical curves lie above the theoretical worst-case upper bound.



(a) Number of participants (b) Empirical Decentralization

Fig. 1: Samples are drawn from Uniform $(0, 1)$ distribution

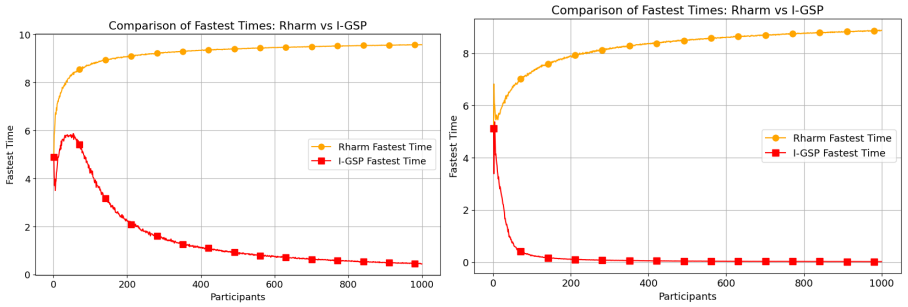


(a) Number of participants (b) Empirical Decentralization

Fig. 2: Samples are drawn from Exponential distribution with parameter $\lambda = 1$ normalized to $(0, 1)$

In the second set of experiments, we again sample agents' costs from Uniform and Exponential distributions and assign times according to the Uniform $[0, 10]$ distribution. To ensure that lower-cost agents take longer to complete the task, we now sample the costs and times separately and sort both lists. We first determine the number of participants using the R^{harm} and then apply $\mathcal{M}^{\text{I-GSP}}$ to select the same number of participants. We record the fastest task completion time

under each mechanism. The results confirm that $\mathcal{M}^{\text{I-GSP}}$ consistently outperforms R^{harm} in terms of the fastest completion time, as expected, since it prioritizes selecting the fastest agents while maintaining feasibility.



(a) Costs drawn from Uniform $(0, 1)$ (b) Costs drawn from Exponential $(\lambda = 1)$

Fig. 3: Times drawn from Uniform $(0, 10)$

6 Conclusion

We introduce a simple model for analyzing the trade-offs between decentralization and efficiency in verifiable computation. We characterize the power and limitations of revelation and non-revelation mechanisms in our model. Our results show provable advantages for revelation mechanisms, i.e., a so-called *revelation gap*.

Our work points to several interesting future avenues. As a first step towards ensuring decentralization and efficiency in verifiable computation, we study the setting where there is a single request/client. The setting where solution providers simultaneously interact with (and budget their computational resources across) multiple clients' mechanisms is left as a future endeavor. A limitation of our model is the assumption that it is not possible for an agent to copy answers from other agents (i.e., claim rewards without doing any work). In the real world, therefore, for our theoretical guarantees to go through, there needs to be a secure and authenticated channel (e.g., answers are signed and encrypted) that the agents can use to interact with the client. Furthermore, we study a specific family of non-revelation mechanisms. Most real-world implementations we are aware of do belong in this family; however, it is an interesting (theoretical and practical) question whether one can bypass our negative results for non-revelation mechanisms by considering different formats, e.g., allowing for more interaction between the agents and the mechanism. Our model assumes that an agent has a fixed cost that they pay to produce a solution at a fixed time; we leave it open to consider a setting where agents can incur a higher cost to provide a faster solution, but we conjecture that incentives have a milder effect in such a setting. Finally, we do not consider potential collusion among the solution providers.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. Alexandros Psomas and Athina Terzoglou are supported in part by an NSF CAREER award CCF-2144208, and a research award from the Herbert Simon Family Foundation.

References

BKDV96. Michael R Baye, Dan Kovenock, and Casper G De Vries. The all-pay auction with complete information. *Economic Theory*, 8:291–305, 1996.

Blo24. Blockchain.com. Bitcoin hashrate distribution, 2024. `https://www.blockchain.com/explorer/charts/pools`.

CKM23. Pyrros Chaidos, Aggelos Kiayias, and Evangelos Markakis. Blockchain participation games. In *International Conference on Web and Internet Economics*, pages 169–187. Springer, 2023.

Cor07. Luis C Corchón. The theory of contests: a survey. *Review of economic design*, 11:69–100, 2007.

DA24. Dune-Analytics. Monthly statistics of zk-pairing, 2024. `https://dune.com/frits/zk-verify-pairing`.

DDR+23. Mohamed Damak, Cihan Duran, Anthony Raziano, Lawrence Wilkinson, Alex Johnston Andrew O'Neill, Dhaval Patel, and Josh Stokesberry. Utility at a cost: Assessing the risks of blockchain oracles. S&P Global, 2023. `https://www.spglobal.com/_division_assets/images/special-editorial/blockchain-oracles/spgmi_utility-at-a-cost-v5.pdf`.

EOS07. Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American economic review*, 97(1):242–259, 2007.

FH18. Yiding Feng and Jason D Hartline. An end-to-end argument in mechanism design (prior-independent auctions for budgeted agents). In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 404–415. IEEE, 2018.

FHL21. Yiding Feng, Jason D Hartline, and Yingkai Li. Revelation gap for pricing from samples. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1438–1451, 2021.

Fla24a. Flashbots. Mev-boost analytics, 2024. `https://www.relayscan.io/overview?t=7d`.

Fla24b. Flashbots. Overview of mev-boost, 2024. `https://docs.flashbots.net/flashbots-mev-boost/introduction`.

FPV15. Eric Friedman, Christos-Alexandros Psomas, and Shai Vardi. Dynamic fair division with minimal disruptions. In *Proceedings of the sixteenth ACM conference on Economics and Computation*, pages 697–713, 2015.

GBE+18. Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert van Renesse, and Emin Gün Sirer. Decentralization in bitcoin and ethereum networks. In Sarah Meiklejohn and Kazue Sako, editors, *Financial Cryptography and Data Security*, pages 439–457, Berlin, Heidelberg, 2018. Springer Berlin Heidelberg. `https://doi.org/10.1007/978-3-662-58387-6_24`.

GGP10. Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *Advances in Cryptology–CRYPTO 2010: 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings 30*, pages 465–482. Springer, 2010. https://link.springer.com/chapter/10.1007/978-3-642-14623-7_25.

GKCC13. Arthur Gervais, Ghassan Karame, Srdjan Capkun, and Vedran Capkun. Is bitcoin a decentralized currency? Cryptology ePrint Archive, Paper 2013/829, 2013. https://eprint.iacr.org/2013/829.

HKFTW23. Lioba Heimbach, Lucianna Kiffer, Christof Ferreira Torres, and Roger Wattenhofer. Ethereum's proposer-builder separation: Promises and realities. In *Proceedings of the 2023 ACM on Internet Measurement Conference*, pages 406–420, 2023. https://doi.org/10.1145/3618257.3624824.

MS01. Benny Moldovanu and Aner Sela. The optimal allocation of prizes in contests. *The American Economic Review*, 91(3):542–558, 2001.

Net24. Gevulot Network. Gevulot, 2024. https://docs.gevulot.com/gevulot-docs/network/overview.

nF24. =nil; Foundation. Proof market, 2024. https://nil.foundation/blog/post/proof-market.

OKK24. Christina Ovezik, Dimitris Karakostas, and Aggelos Kiayias. Sok: A stratified approach to blockchain decentralization. In *Financial Cryptography and Data Security*, Lecture Notes in Computer Science. Springer, 2024. https://fc24.ifca.ai/.

PHGR16. Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. *Communications of the ACM*, 59(2):103–112, 2016. https://doi.org/10.1145/2856449.

PLD23. Amirmohammad Pasdar, Young Choon Lee, and Zhongli Dong. Connect api with blockchain: A survey on blockchain oracle implementation. *ACM Comput. Surv.*, 55(10), feb 2023.

Sta25. Statista. Public cloud - worldwide, 2025. https://www.statista.com/outlook/tmo/public-cloud/worldwide.

suc24. Succinct network: Prove the world's software, 2024.

Sup23. Supra Research. Supra vrf service: Whitepaper. Supra dVRF Product Documentation, 2023. https://supra.com/docs/SupraOracles-VRF-Service-Whitepaper.pdf.

Swe22. Craig Sweney. Chainlink verifiable random function activity surges. Delphi Digital Report, 2022. https://members.delphidigital.io/reports/chainlink-verifiable-random-function-activity-surges.

tai24. taiko.xyz. Taiko tokenomics, 2024. https://github.com/taikoxyz/taiko-mono/blob/alpha-4/packages/protocol/docs/tokenomics_staking.md.

WEY+24. Anton Wahrstätter, Jens Ernstberger, Aviv Yaish, Liyi Zhou, Kaihua Qin, Taro Tsuchiya, Sebastian Steinhorst, Davor Svetinovic, Nicolas Christin, Mikolaj Barczentewicz, and Arthur Gervais. Blockchain censorship. In *Proceedings of the ACM on Web Conference 2024*, page 1632–1643, 2024.

WZY+25. Wenhao Wang, Lulu Zhou, Aviv Yaish, Fan Zhang, Ben Fisch, and Benjamin Livshits. Mechanism design for zk-rollup prover markets. In *29th International Conference on Financial Cryptography and Data Security*. Springer, 2025.

A Missing proofs

Useful technical facts. We state two lemmas that will be useful throughout the missing proofs. The first one is from [FPV15].

Lemma 1 ([FPV15]). *For natural numbers $b > a > 1$,*

$$\ln\left(\frac{b}{a-1}\right) - \frac{1}{2a-2} \leq \sum_{j=a}^{b} \frac{1}{j} \leq \ln\left(\frac{b}{a-1}\right).$$

Lemma 2. *For m non-negative real numbers, $0 \leq x_1 \leq x_2 \leq \ldots \leq x_m$ such that $\sum_{i=1}^{m} x_i \leq 1$, it holds that for all $i \in [m]$: $x_i \leq \frac{1}{m-i+1}$.*

Proof (Lemma 2). We have that $1 \geq \sum_{k=i}^{m} x_k = x_m + x_{m-1} + \cdots + x_i \geq x_i + x_i + \cdots + x_i = (m-i+1)x_i$. Therefore, $x_i \leq \frac{1}{m-i+1}$. □

Missing proofs.

Proof (Claim 1). For the sake of contradiction, assume that the k-best set S_k^* is not $\frac{k}{k^*}$-efficient, i.e. $\min_{i \in S_k^*} t_i > t^*_{\frac{k}{k^*}}$. By Definition 3, there exists a set that is $\frac{k}{k^*}$-decentralized and $\frac{k}{k^*}$-efficient, let that be $\hat{S}$. Let $i^* \in \hat{S}$ be the index that $t_{i^*} = t^*_{\frac{k}{k^*}}$. Since $\hat{S}$ is a feasible set, we have that $c_{i^*} + \sum_{j \in \hat{S} \setminus \{i^*\}} c_j \leq 1$. By construction of S_k^*, since $i^* \notin S_k^*$ it must be that $c_{i^*} + \sum_{j=1}^{k-1} c_j > 1$ otherwise it would be in S_k^*. That contradicts the feasibility of $\hat{S}$, since $\sum_{j=1}^{k-1} c_j$ is the cost of the cheapest set of size $k-1$. □

Proof (Proposition 1). It suffices to show the proposition for $\alpha \in \{\frac{1}{k^*}, \frac{2}{k^*}, \cdots, 1\}$, where αk^* is an integer. Let $S^*_{\alpha+\frac{1}{k^*}}$ be the set that achieves $t^*_{\alpha+\frac{1}{k^*}}$, i.e., $S^*_{\alpha+\frac{1}{k^*}} \in \min_{S:|S| \geq (\alpha+\frac{1}{k^*})k^*, \sum_{i \in S} c_i \leq 1} \min_{i \in S} t_i$. Let $S = S^*_{\alpha+\frac{1}{k^*}} \setminus \{\arg\max_{i \in S^*_{\alpha+\frac{1}{k^*}}} t_i\}$. The set S is a of size αk^* since $|S| = \alpha k^*$ with cost $\sum_{i \in S} c_i < \sum_{i \in S^*_{\alpha+\frac{1}{k^*}}} c_i \leq 1$. Therefore $t^*_\alpha \leq \min_{i \in S} t_i = \min_{i \in S^*_{\alpha+1}} t_i = t^*_{\alpha+1}$. □

Proof (Theorem 4). Assume w.l.o.g. that $c_i \leq c_{i+1}$ for all i, and let $m_z = \arg\min_{a \in \mathbb{N}} \left\{\sum_{x=a}^{a+z-1} \frac{1}{x} \leq 1\right\}$.

First, we prove that a pure Nash equilibrium exists via induction on n. For $n = 1$ the statement is trivial. Assume that a pure Nash equilibrium exists for agents with costs $c_1 \leq \cdots \leq c_{n-1}$, and let S^* be the set of agents participating in this equilibrium, where $\ell = |S^*|$. Let $rank(i, S)$ be the ranking of agent i among a subset S of agents, where $rank(i, S) = 1$ means that t_i is the smallest among agents in S, and $rank(i, S) = |S|$ means that t_i is the largest among agents in S. For all $i = 1, \ldots, n-1$ such that $i \notin S^*$, we know that c_i is strictly larger than the reward they get when they participate in $S^* \cup \{i\}$, i.e. $c_i > \frac{1}{m_{\ell+1} + rank(i, S^* \cup \{i\}) - 1}$. Consider an instance with costs $c_1 \leq \cdots \leq c_n$. If S^* is not an equilibrium, it must be that agent n wants to join, i.e. $c_n \leq \frac{1}{m_{\ell+1}}$. If

$S^* \cup \{n\}$ is not an equilibrium, it must be that an agent in S^* is not adequately rewarded (i.e., it is not the case that an additional agent wants to join); let i^* be the highest cost agent in S^* that is not adequately rewarded. Finally, $S^* \cup \{n\} \setminus \{i^*\}$ must be an equilibrium: (i) agent n gets reward $\frac{1}{m_\ell} \geq \frac{1}{m_{\ell+1}} \geq c_n$, (ii) agent $i \in S^* \setminus \{i^*\}$ either gets reward at least her reward in S^* if her rank didn't change or, otherwise, her rank did change compared to $rank(i, S^*)$, but her cost is at most her reward in $S^* \cup \{n\}$ (since i^* has the highest cost and unhappy agent) (iii) agent $i \notin S^* \cup \{n\} \setminus \{i^*\}$ has a cost strictly larger than her reward if she joins, since $rank(i, S^* \cup \{i\}) \leq rank(i, S^* \cup \{n, i\} \setminus \{i^*\})$.

Next, we prove the decentralization bound. Consider a pure Nash equilibrium of $\mathcal{R}^{\text{Harm}}$ with ℓ agents participating. When an additional agent submits a solution, her reward is at least $\frac{1}{m_{\ell+1}}$; therefore, for all i such that $a_i = \perp$, $c_i > \frac{1}{m_{\ell+1}}$. Let i^* be the agent with the smallest cost among those not participating, noting that $i^* \leq (1 - 1/e)k^* - 8$, since we sorted in increasing cost, and since, otherwise, the theorem follows immediately. Since $\sum_{i=1}^{k^*} c_i \leq 1$, Lemma 2 implies that $\frac{1}{k^* - i^* + 1} \geq c_{i^*} > \frac{1}{m_{\ell+1}}$. Re-arranging we have

$$m_{\ell+1} > k^* - i^* + 1 \geq \frac{k^*}{e} + 9. \tag{1}$$

We know that $m_{\ell+1}$ is the smallest non-negative integer such that $\sum_{x=m_{\ell+1}}^{m_{\ell+1}+\ell} \frac{1}{x} \leq 1$. Note that this does not imply that $\sum_{x=m_{\ell+1}}^{m_{\ell+1}+\ell+1} \frac{1}{x} > 1$; for example, for $\ell + 1 = 3$, $1/2 + 1/3 + 1/4 > 1$, $1/3 + 1/4 + 1/5 < 1$ (so, $m_3 = 3$), but also $1/3 + 1/4 + 1/5 + 1/6 < 1$ (that is, $m_4 = 3$). However, it is true that $\sum_{x=m_{\ell+1}+\ell-1}^{m_{\ell+1}+\ell-1} \frac{1}{x} > 1$. The next claim shows that we can always add a constant number of terms after $\frac{1}{m_{\ell+1}+\ell}$ so that $\sum_{x=m_{\ell+1}}^{m_{\ell+1}+\ell+c} \frac{1}{x} > 1$; this, in turn, will give us an upper bound on $m_{\ell+1}$ that we can combine with Equation (1).

Claim 10. *For all $m_{\ell+1} \geq 2$, $\sum_{x=m_{\ell+1}}^{m_{\ell+1}+\ell+20} \frac{1}{x} > 1$.*

Proof. Since $\sum_{x=m_{\ell+1}+\ell-1}^{m_{\ell+1}+\ell-1} \frac{1}{x} > 1$, it suffices to show that $\frac{1}{m_{\ell+1}-1} < \sum_{x=m_{\ell+1}+\ell}^{m_{\ell+1}+\ell+c} \frac{1}{x}$.

We know that $1 \geq \sum_{x=m_{\ell+1}}^{m_{\ell+1}+\ell} \frac{1}{x} \geq \ln(m_{\ell+1} + \ell) - \ln(m_{\ell+1} - 1) - \frac{1}{2m_{\ell+1}-2} \geq \ln(m_{\ell+1} + \ell) - \ln(m_{\ell+1} - 1) - \frac{1}{2}$, where the second inequality uses Lemma 1 and the third inequality uses the fact that $m_{\ell+1} \geq 2$. Re-arranging we have $\frac{m_{\ell+1}+\ell}{m_{\ell+1}-1} \leq e^{1.5}$, or $\frac{1}{m_{\ell+1}} \leq \frac{e^{1.5}}{m_{\ell+1}+\ell} \leq \frac{5}{m_{\ell+1}+\ell}$. Therefore, it suffices to show that $\sum_{x=m_{\ell+1}+\ell}^{m_{\ell+1}+\ell+c} \frac{1}{x} \geq \frac{5}{m_{\ell+1}+\ell}$, or $\sum_{x=m_{\ell+1}+\ell+1}^{m_{\ell+1}+\ell+c} \frac{1}{x} \geq \frac{4}{m_{\ell+1}+\ell}$.

Using Lemma 1 we have that $\sum_{x=m_{\ell+1}+\ell+1}^{m_{\ell+1}+\ell+c} \frac{1}{x} \geq \ln\left(\frac{m_{\ell+1}+\ell+c}{m_{\ell+1}+\ell}\right) - \frac{1}{2(m_{\ell+1}+\ell)}$. Re-arranging, and replacing $z = m_{\ell+1} + \ell$ (to simplify notation) and plugging in $c = 20$, it suffices to show that

$$\ln\left(1 + \frac{c}{z}\right) = \ln\left(1 + \frac{20}{z}\right) \geq \frac{4.5}{z}.$$

One can confirm that this expression is true for $z = 2, 3, 4, 5$ by doing simple calculations. For $z \geq 5$ we use the lower bound $\ln(1+x) \geq \frac{x}{1+x}$ to get $\ln\left(1 + \frac{20}{z}\right) \geq \frac{20}{20+z}$, which is at least $\frac{4.5}{z}$ for all $z \geq 6$. This confirms the proof of Claim 10. □

Using Claim 10 and Lemma 1 we have $1 < \sum_{x=m_{\ell+1}}^{m_{\ell+1}+\ell+20} \frac{1}{x} \leq \ln\left(\frac{m_{\ell+1}+\ell+20}{m_{\ell+1}-1}\right)$, which implies $m_{\ell+1} \leq \frac{\ell+20+e}{e-1}$. Combining with Equation (1) we have $\frac{\ell+20+e}{e-1} > \frac{k^*}{e} + 8$, or $\ell \geq \left(1 - \frac{1}{e}\right) k^* + 9(e-1) - (20+e) \geq \left(1 - \frac{1}{e}\right) k^* - 8$, concluding the proof of Theorem 4. □

Proof (Theorem 5). Let $\mathcal{R}$ be a non-revelation and reward-monotone mechanism that is α-decentralized. Fix k^*; we will consider instances with $n = k^*$ agents. Let $\ell^* = \lceil \alpha k^* \rceil$ be the number of agents that the mechanism needs to incentivize to participate in order to satisfy α-decentralization. We will consider the behavior of $\mathcal{R}$ when it receives solutions only at time T, noting that, similar to the proof of Theorem 2, submitting a solution at time T and not submitting a solution are the only options for agents whose types are (c, T). Specifically, let $r_i^{(z)}$ be the i-th smallest reward when z agents submit solutions (at time T), where $z = \ell^*, \ell^* + 1, \ldots, k^*$.

First, notice that, for some z, the smallest reward must be at least $1/k^*$, otherwise when all agents have costs $1/k^*$, the agent receiving the smallest reward wants to deviate. Similarly, the second smallest reward must be at least $1/(k^*-1)$, and so on. This is formalized in the following claim.

Claim 11. *For all $i = 1, \ldots, \ell^*$, $\min_z r_i^{(z)} \geq \frac{1}{k^*+1-i}$.*

Proof. Assume that for some i $\min_z r_i^{(z)} = \frac{1}{k^*+1-i} - \delta$, for some $\delta > 0$. Consider an equilibrium for the case where k^*+1-i agents have types $(\frac{1}{k^*+1-i} - \frac{\delta}{n}, T)$ and $i-1$ agents have types $(\frac{\delta}{n}, T)$; the number of agents participating in this equilibrium is at least $\ell^* \geq i$. The i-th smallest reward is at most $\frac{1}{k^*+1-i} - \delta < \frac{1}{k^*+1-i} - \frac{\delta}{n}$, and therefore one of the agents with cost $\frac{1}{k^*+1-i} - \frac{\delta}{n}$ (the i-th smallest cost) cannot be rewarded adequately, contradicting the equilibrium condition. □

Since $\mathcal{R}$ is reward-monotone, Claim 11 implies that $r_i^{(\ell^*)} \geq \frac{1}{k^*+1-i}$. Since $r_1^{(\ell^*)}, \ldots, r_{\ell^*}^{(\ell^*)}$ must be a feasible set of rewards, we have

$$1 \geq \sum_{i=1}^{\ell^*} r_i^{(\ell^*)} \geq \sum_{i=1}^{\ell^*} \frac{1}{k^*+1-i} = \sum_{i=k^*+1-\ell^*}^{k^*} \frac{1}{i}. \tag{2}$$

Using the fact that $\ln(b) + 1 \geq H_b \geq \ln(b)$, we can lower bound the RHS of Equation (2) by $\ln(k^*) - \ln(k^* - \ell^*) - 1$; re-arranging and simplifying we have $k^* - \ell^* \geq k^*/e^2$. Using this bound we can apply Lemma 1 to lower bound the RHS of Equation (2) as $\ln(k^*) - \ln(k^* - \ell^*) - \frac{1}{2(k^*-\ell^*)} \geq \ln(k^*) - \ln(k^* - \ell^*) - \frac{e^2}{2k^*}$. Re-arranging and simplifying we have $\ell^* \leq \left(1 - \frac{1}{e^{1+\frac{e^2}{2k^*}}}\right) k^*$. □

Proof (Theorem 7). Let $(c_1, t_1), \ldots, (c_n, t_n)$ be the underlying instance. It is easy to see that no agent has an incentive to delay their submission, since by delaying she can only be miscategorized into a slower bucket and receive a smaller reward. Therefore, for all $i \in [n]$, $a_i \in \{t_i, \perp\}$, and thus we only need to consider as actions "participate" or "not participate".

First, we show that a pure Nash equilibrium exists. Let $\hat{S}$ be the k-best set of $\mathcal{R}^{\text{Best-Set}}$ when all agents participate. Let ℓ be the index of the fastest agent in $\hat{S}$. When only the agents in $\hat{S}$ participate, this is an equilibrium. To see this, consider any agent $i \in \hat{S}$; she has no incentive to deviate since she is rewarded according to her profile and cannot submit a solution in a faster (and more expensive) bucket. Now, consider any agent $i \notin \hat{S}$. There are four cases that we need to examine about the bucket $b(t_i)$. Let j^* be the bucket of the fastest agent in $\hat{S} \setminus \{\ell\}$(fastest among the slow solutions). If $b(t_i) > b(t_\ell)$, it must be that $c(t_i) + \sum_{\hat{S}\setminus\{\ell\}} c_j > 1$, hence the agent prefers to not participate. If $b(t_i) = j^*$ it must be that $t_i > \min_{z\in\hat{S}\setminus\{\ell\}} t_z$ (i.e., not the fastest in the bucket) since $i \notin \hat{S}$. Since agents cannot submit faster solutions, there is no way to be included in the set, thus she does not want to participate. If $b(t_i) = b(t_\ell)$, the same argument applies since i is in the same bucket as an agent in $\hat{S}$ but not the fastest in the bucket. Finally, for $i \notin \hat{S}$ such that $j^* < b(t_i) < b(t_\ell)$ (not in a rewarding bucket), notice that if i participates, $\hat{S}$ will not change since i is neither in the cheapest bucket nor is the fastest agent available.

We will show that the above equilibrium is unique. First, it is easy to see that there is no equilibrium of size $k+1$, since at most k agents get rewards. Next, for any set of submitted solutions, every one of the $k-1$ "cheapest and fastest" agents is always incentivized to also submit a solution, therefore, all equilibria must include all of the $k-1$ "cheapest and fastest" agents. Therefore, we cannot have an equilibrium of size $k-1$ either, since agent $\ell = \arg\min_i\{t_i : \sum_{j\in S^{(1)}\cup S^{(2)}} c_j \leq 1\}$ prefers to join. Finally, in a set of solutions with the $k-1$ cheapest agents and another agent that is not ℓ cannot be an equilibrium, since agent ℓ prefers to join. $\frac{k}{k^*}$-decentralization is immediately implied. To conclude the proof, we need to show that the rule is also $\frac{k}{k^*}$-efficient, which is an implication of Claim 1. □

Claim 12. *The inverse k-price auction of Example 2 is* $1/2$ *decentralized.*

Proof. Assume that reported costs satisfy $c_1 \leq c_2 \cdots \leq c_n$, and let k be the largest integer such that $k \cdot c_{k+1} \leq 1$. We know that, for k^*, $\sum_{i=1}^{k^*} c_i \leq 1$, therefore, Lemma 2 then implies that $c_i \leq \frac{1}{k^*-i+1}$ for all $i \leq k^*$. By the choice of k we have $1 < (k+1)\cdot c_{k+2} \leq \frac{k+1}{k^*-(k+2)+1}$. Re-arranging we have $k \geq \frac{k^*}{2} - 1$. □

Proof (Theorem 8). Suppose that there is a revelation mechanism $\mathcal{M}$ that achieves $(1/2+\epsilon)$-decentralization, for some $\epsilon > 0$. Consider an instance with $n = 2$ agents and decentralization factor $k^* = 2$: let $\mathcal{I} = ((c_1, t_1), (c_2, t_2))$, where $c_1 + c_2 < 1$. Now, consider the following two instances $\mathcal{I}_1 = ((\widehat{c_1}, t_1), (c_2, t_2))$ and $\mathcal{I}_2 = ((c_1, t_1), (\widehat{c_2}, t_2))$ such that $\widehat{c_1} + c_2 = 1$ and $c_1 + \widehat{c_2} = 1$, respectively. By assumption, $\mathcal{M}$ must incentivize both agents to submit solutions in all instances, i.e. the sum of allocations must be equal to 2. By individual rationality, it must

then be that $r^{\mathcal{M}}((\widehat{c_1}, t_1), (c_2, t_2)) = (\widehat{c_1}, c_2)$ and $r^{\mathcal{M}}((c_1, t_1), (\widehat{c_2}, t_2)) = (c_1, \widehat{c_2})$. Now, if $r_1^{\mathcal{M}}((c_1, t_1), (c_2, t_2)) < \widehat{c_1}$, agent 1 has an incentive to deviate to reporting $(\widehat{c_1}, t_1)$. Similarly, if $r_2^{\mathcal{M}}((c_1, t_1), (c_2, t_2)) < \widehat{c_2}$ agent 2 is incentivized to report $(\widehat{c_2}, t_2)$. Combining the above inequalities, we have that $r_1^{\mathcal{M}}((c_1, t_1), (c_2, t_2)) \geq \widehat{c_1}$ and $r_2^{\mathcal{M}}((c_1, t_1), (c_2, t_2)) \geq \widehat{c_2}$. However, $c_1 < \widehat{c_1}$, since $c_1 + c_2 < 1$, and $\widehat{c_1} + c_2 = 1$. Thus, $r_1^{\mathcal{M}}((c_1, t_1), (c_2, t_2)) + r_2^{\mathcal{M}}((c_1, t_1), (c_2, t_2)) \geq \widehat{c_1} + \widehat{c_2} > 1$; a contradiction. □