On the (In)approximability of Combinatorial Contracts

Tomer Ezra
Simons Laufer Mathematical Sciences Institute, Berkeley, CA, USA

Michal Feldman
Tel Aviv University, Israel

Maya Schlesinger
Tel Aviv University, Israel

Abstract

We study two recent combinatorial contract design models, which highlight different sources of complexity that may arise in contract design, where a principal delegates the execution of a costly project to others. In both settings, the principal cannot observe the choices of the agent(s), only the project's outcome (success or failure), and incentivizes the agent(s) using a contract, a payment scheme that specifies the payment to the agent(s) upon a project's success. We present results that resolve open problems and advance our understanding of the computational complexity of both settings.

In the *multi-agent* setting, the project is delegated to a team of agents, where each agent chooses whether or not to exert effort. A success probability function maps any subset of agents who exert effort to a probability of the project's success. For the family of submodular success probability functions, Dütting et al. [2023] established a poly-time constant factor approximation to the optimal contract, and left open whether this problem admits a PTAS. We answer this question on the negative, by showing that no poly-time algorithm guarantees a better than 0.7-approximation to the optimal contract. For XOS functions, they give a poly-time constant approximation with value and demand queries. We show that with value queries only, one cannot get any constant approximation.

In the *multi-action* setting, the project is delegated to a single agent, who can take any subset of a given set of actions. Here, a success probability function maps any subset of actions to a probability of the project's success. Dütting et al. [2021a] showed a poly-time algorithm for computing an optimal contract for gross substitutes success probability functions, and showed that the problem is NP-hard for submodular functions. We further strengthen this hardness result by showing that this problem does not admit any constant factor approximation. Furthermore, for the broader class of XOS functions, we establish the hardness of obtaining a $n^{-1/2+\varepsilon}$-approximation for any $\varepsilon > 0$.

2012 ACM Subject Classification Theory of computation → Algorithmic game theory

Keywords and phrases algorithmic contract design, combinatorial contracts, moral hazard

Digital Object Identifier 10.4230/LIPIcs.ITCS.2024.44

Related Version *Full Version*: https://arxiv.org/abs/2311.18425

Funding Feldman and Schlesinger received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 866132), by an Amazon Research Award, and by the NSF-BSF (grant number 2020788). Ezra was funded by the National Science Foundation under Grant No. DMS-1928930 and by the Alfred P. Sloan Foundation under grant G-2021-16778, while the author was in residence at the Simons Laufer Mathematical Sciences Institute (formerly MSRI) in Berkeley, California, during the Fall 2023 semester.

1 Introduction

Contract theory is a pillar in microeconomics, studying how to incentivize agents to exert costly effort when their actions are hidden. This problem is explored using the principal-agent model introduced by Holmström [17] and Grossman and Hart [14]. In this model, a principal wishes to delegate the execution of a costly task to an agent who can take one of n actions, each associated with a cost and a probability distribution over outcomes. The agent's action is hidden from the principal, who can observe only the realized outcome. To incentivize the agent to exert effort, the principal designs a contract, which is a payment scheme that specifies a payment for every possible outcome. The goal of the principal is to find a contract that maximizes her utility (expected reward minus expected payment), assuming the agent takes the action that maximizes his own utility (expected payment minus cost). This problem can be solved in polynomial time using linear programming [14].

In recent years, the principal-agent model has been extended to combinatorial settings along different dimensions, such as multiple agents [2, 12, 9], multiple actions [8] and exponentially many outcomes [11].

In this work, we study two of these combinatorial contract models, namely the multi-agent and multi-action settings. In both of these models, the focus is on the case of binary outcome, where the project can either succeed or fail, and the principal receives some reward (normalized to 1) if the project succeeds. Notably, finding an (approximate) optimal contract in the binary-outcome model, is equivalent to finding an (approximate) optimal linear contract in settings with more than two outcomes. Thus, we restrict attention to linear contracts without loss of generality. Moreover, since our focus is on hardness results, restricting attention to the binary-outcome case only strengthens our results.

Setting 1: Multi-agent. In the multi-agent model [2, 9], the principal delegates the execution of a costly project to a team of n agents. Every agent can either exert effort (at some cost to the agent) or not. At the heart of the model is a success probability function $f : 2^{[n]} \to [0, 1]$, which specifies, for every subset of agents who exert effort, the probability that the project succeeds.

The principal incentivizes the agents through a contract that specifies for every agent i, a non-negative payment α_i that the principal pays the agent if the project succeeds. The principal's utility is defined as her expected reward minus the expected sum of payments to the agents. Given a contract, an agent's utility is defined as his expected payment from the contract minus his cost if he chooses to exert effort. Thus, a contract by the principal induces a game between the agents, and we consider the agent actions in an equilibrium of the game. The principal's goal is to maximize her expected utility in equilibrium.

For *submodular* success probability functions, Dütting et al. [9] devise a constant factor approximation algorithm, using value query access (a value query receives a set $S \subseteq [n]$ and returns the value $f(S)$). They left as an open problem whether the problem admits a PTAS. For the larger class of *XOS* success probability functions, they also give a constant approximation algorithm, using both value and demand queries (a demand query receives a price vector $p \in \mathbb{R}^n_{\geq 0}$, and returns a set S that maximizes $f(S) - \sum_{i \in S} p_i$). For the XOS class, they show that it is not possible to obtain a better-than-constant approximation with value and demand queries.

Setting 2: Multi-action. In the multi-action model [8], the principal delegates the execution of the project to a single agent, who can take any subset of n possible actions. Each action i is associated with a cost c_i, and when the agent executes a set of actions $S \subseteq [n]$, he incurs the sum of their costs. Here, the success probability function $f : 2^{[n]} \to [0, 1]$ maps any subset of the actions to a success probability of the project.

In this model, the principal specifies a single non-negative payment α that is paid to the agent if the project succeeds. The agent then chooses a subset of actions that maximizes his utility (the expected payment from the principal minus the cost he incurs). The principal's utility is the expected reward minus the expected payment to the agent.

Dütting et al. [8] show that computing an optimal contract for submodular success probability functions is NP-hard, and left as an open question whether there exists an approximation algorithm for the problem, for submodular success probability functions, as well as for the larger classes of XOS and subadditive functions. (For the class of gross-substitutes success probability functions – a strict subclass of submodular functions – they devise a polytime algorithm for computing an optimal contract, using access to a value oracle.)

1.1 Our Results

Setting 1: Multi-agent. Our first set of results concern the multi-agent setting. The first result resolves the open question from [9] in the negative, showing that the multi-agent problem with submodular success probability functions does not admit a PTAS.

▶ **Theorem** (multi-agent, submodular). *In the multi-agent model, with submodular success probability function, no polynomial time algorithm with value oracle access can approximate the optimal contract to within a factor of* 0.7*, unless P=NP.*

We then turn to XOS success probability functions. Dütting et al. [9] provide a poly-time constant approximation for XOS with demand queries. We show that no algorithm can do better than $O(n^{-1/6})$-approximation with poly-many value queries, thus establishing a separation between the power of value and demand queries for XOS functions.

▶ **Theorem** (multi-agent, XOS). *In the multi-agent model, with XOS success probability function, no (randomized) algorithm that makes poly-many value queries can approximate the optimal contract (with high probability) to within a factor greater than* $4n^{-1/6}$.

Setting 2: Multi-action. Our second set of results consider the multi-action model. We first show that obtaining any constant approximation for submodular functions is hard.

▶ **Theorem** (multi-action, submodular). *In the multi-action model, with submodular success probability function* f*, no polynomial time algorithm with value oracle access can approximate the optimal contract to within a constant factor, unless P=NP.*

We then show that for the broader class of XOS success probability functions, it is hard to obtain a $n^{-1/2+\varepsilon}$-approximation for any $\varepsilon > 0$.

▶ **Theorem** (multi-action, XOS). *In the multi-action model, with XOS success probability function, under value oracle access, for any* $\varepsilon > 0$*, no polynomial time algorithm with value query access can approximate the optimal contract to within a factor of* $n^{-\frac{1}{2}+\varepsilon}$*, unless P=NP.*

1.2 Our Techniques

Submodular functions. Both of our hardness results for submodular functions are based on an NP-hard promise problem for normalized unweighted coverage functions (a subclass of submodular), which is a generalization of a result by Feige [13]. We introduce this problem and Section 3, and the proof of its hardness can be found in the full version of this paper.

In particular, we show that it is NP-hard to distinguish between a normalized unweighted coverage function f that has a relatively small set S with $f(S) = 1$, and one for which a significantly larger set would be required to get close to 1. This leads to our submodular hardness results as follows: by setting uniform costs to all actions/agents, an optimal contract has a significantly different utility to the principal in each case. When a relatively small set

S achieves $f(S) = 1$, the principal's utility is relatively high, and when a significantly larger set is required to get close to 1, the principal's utility is relatively low. In the multi-agent setting, the principal's utility from a given contract is easy to compute, which allows an approximately optimal contract to distinguish between the two cases.

In the multi-action setting, where the principal's utility from a given contract isn't necessarily easy to compute, our reduction also involves the addition of a new action. This new action is defined in such a way that only a high payment to the agent incentivizes the agent to take it. Thus - the principal has to decide whether to choose the optimal contract for the original problem, or to incentivize the agent to take this new action, which leads to a separation of approximately optimal contracts between the two cases.

Multi-action, XOS functions. For this result, we construct a reduction from the problem of approximating the size of the largest clique in a graph to our problem. We rely on the hardness result of Håstad [16] and Zuckerman [25] for approximating the largest clique in a graph G, denoted by $\omega(G)$. In particular, Håstad [16] and Zuckerman [25] show that there is no poly-time algorithm that approximates $\omega(G)$ within a factor of $n^{-1+\varepsilon}$ (for any $\varepsilon > 0$) unless P=NP. We show that given a β-approximation algorithm (for $\beta \in (0, 1)$) for the optimal contract, one can approximate $\omega(G)$ within a factor of $\beta^2/4$, which implies our inapproximability of $n^{-1/2+\varepsilon}$ for the optimal contract. To achieve this, for any parameter $\beta \in (0, 1)$, we give an algorithm that on input (G, δ), where G is a graph and $\delta \in \mathbb{N}^+$, creates an instance of the multi-action contract problem with an XOS success probability function, for which value queries can be computed in polynomial time. In the constructed instance there are only two "reasonable" candidates for a contract, regardless of the structure of G; these are the values of α at which the agent's best response may change. The lower of these candidates is better than the other by at least a factor of β when $\omega(G) \leq \delta$, and the reverse is true when $\omega(G) \geq 2\delta/\beta^2$. This gives us the ability to distinguish between the case where $\omega(G) \leq \delta$ and $\omega(G) \geq 2\delta/\beta^2$. By repeating this process for different values of δ we can approximate $\omega(G)$ within a factor of $\beta^2/4$.

Multi-agent, XOS functions. Our inapproximability result for this case is information theoretic, and relies on "hiding" a good contract, so that no algorithm with poly-many value queries can find it with non-negligible probability. In particular, for any n, we choose a set $G \subseteq A$ of $m = n^{1/3}$ "good" agents uniformly at random. We define an XOS success probability function such that sets of size $O(m)$ may have a high success probability only if they have a large intersection with G, and any value query reveals negligible information regarding the set G. We set equal costs such that incentivizing more than $2m$ agents becomes unprofitable to the principal. Thus, in order to get a good approximation, the algorithm must find a relatively small set of agents that has a large intersection with G. Since our construction of the success probability function is such that value queries reveal negligible information on G, the algorithm has a negligible probability of finding such a set.

1.3 Related Work

Multi-agent settings: additional related work. Babaioff et al. [2] introduced a multi-agent model where every agent decides whether to exert effort or not, and succeeds in his own task (independently) with a higher probability if he exerts effort. The project's success is then a function of the individual outcomes by the agents. They show that computing the optimal contract in this model is #P-hard in general, and provide a polytime algorithm for the special case where the project succeeds iff all agents succeed in their individual tasks (AND function). Emek and Feldman [12] show that computing the optimal contract in the special case where the project succeeds iff at least one agent succeeds (OR function) is NP-hard, and provide an FPTAS for this problem.

Dütting et al. [9] extend the model of [2] to the model presented in our paper, where the project's outcome is stochastically determined by the set of agents who have exerted effort, according to a success probability function $f : 2^{[n]} \to [0, 1]$. Their primary result is the development of a constant approximation algorithm for XOS success probability functions. They complement this result by showing an upper bound of $O(1/\sqrt{n})$ for subadditive functions, and an upper bound of a constant for XOS functions. They also show that the problem is NP-hard even for additive functions, and devise an FPTAS for this case.

Vuong et al. [24] study the model of [9] for the case where the function f is supermodular, and show that no polynomial time algorithm can achieve any constant approximation nor an additive FPTAS. They also present an additive PTAS for a special case of graph-based super modular valuations.

Castiglioni et al. [6] study a multi-agent setting in which each agent has his own outcome, which is observable by the principal, and the principal's reward depends on all the individual outcomes. When the principal's reward is supermodular, they show that it is NP-hard to get any constant approximation to the optimal contract. They also give a poly-time algorithm for the optimal contract in special cases. When the principal's reward is submodular, they show that for any $\alpha \in (0, 1)$ it is NP-hard to get a $n^{\alpha-1}$-approximation, and they also provide a poly-time algorithm that gives a $(1 - \frac{1}{e})$-approximation up to a small additive loss.

Dasartha et al. [7] consider a multi-agent setting with graph-based reward functions, and continuous effort levels, and characterize the optimal equilibrium induced by a linear contract.

Multi-action settings: additional related work. Vuong et al. [24] and Dütting et al. [10] further explore the multi-action model of [8]. They present a poly-time algorithm for computing the optimal contract for any class of instances that admits an efficient algorithm for the agent's demand and poly-many "breakpoints" in the agent's demand. A direct corollary of this result is a polynomial time algorithm for computing the optimal contract when the success probability function is supermodular and the cost function is submodular. Dütting et al. [10] further show a class of XOS success probability functions (matching-based) which admits an efficient algorithm for the agent's demand, but has a super-polynomial number of breakpoints in the agent's demand. Computing the optimal contract for this class remains an open problem. (Pseudo) polynomial algorithms are presented for two special cases.

Additional combinatorial contract models. Beyond multi-agent and multi-action, one can consider other dimensions in which a contracting problem grows. For example, Dütting et al. [11] consider a setting with exponentially many outcomes. They show that under a constant number of actions, it is NP-hard to compute an optimal contract. They proceed to weaken their restriction on contracts, and consider "approximate-IC" contracts, in which the principal suggests an action for the agent to take (in addition to the payment scheme), and the agent takes it as long as its utility is not much lower than that of another possible action. They present an FPTAS that computes an approximate-IC contract that gives the principal an expected utility of at least that achieved in the optimal (IC) contract. For an arbitrary number of actions, they show NP-hardness of any constant approximation, even for approximate-IC contracts.

Contracts for agents with types. Guruganesh et al. [15] consider a setting where in addition to hidden action, the agent also has a private type, which changes the effect of each action he takes on the project's outcome. In the private type setting the principal may wish to incentivize the agent with a menu of contracts, i.e., a set of contracts from which the agent is free to choose whichever contract he prefers. They show APX-hardness of both the optimal contract and the optimal menu of contracts. In contrast, Alon et al. [1] consider the case where the agent has a single-dimensional private type, and they present a characterization of implementable allocation rules (mappings of agent types to actions), which allows them to design a poly-time algorithm for the optimal contract with a constant number of actions.

Castiglioni et al. [5] study the case where the agent's private type is Bayesian, i.e., drawn from some known finitely-supported distribution. They study menus of randomized contracts (defined as distributions over payment vectors), wherein upon the agent's choice of randomized contract, the principal draws a single deterministic contract from the distribution, and the agent plays his best response to this deterministic contract. They show that an almost optimal menu of randomized contracts can be computed in polynomial time. They also show that the problem of computing an optimal menu of deterministic contracts cannot be approximated within any constant factor in polynomial time, and that it does not admit an additive FPTAS.

Optimizing the efforts of others. Contract design is part of an emerging frontier in algorithmic game theory regarding optimizing the effort of others (see, e.g., the STOC 2022 TheoryFest workshop with the same title). In addition to contract design, this field includes recent work on strategic classification [19, 3], delegation [18, 4], and scoring rules [23, 21].

2 Model and Preliminaries

We first describe the basic version of the contract design problem, also known as the hidden-action or principal-agent setting. We then present two extensions, one with multiple agents, the other with multiple actions. For simplicity, we restrict attention to a binary-outcome setting, where a project either succeeds or fails.

2.1 Basic Principal-Agent Setting

A single principal interacts with a single agent, in an effort to make a project succeed. The agent has a set A of possible actions, each with associated cost $c_i \geq 0$ and probability $p_i \in [0, 1]$. When the agent selects action $i \in A$, he incurs a cost of c_i, and the project succeeds with probability p_i, and fails with probability $1 - p_i$. If the project succeeds, the principal gets a reward which we normalize to 1. The principal is not aware of which action the agent has taken, only if the project has succeeded or failed.

Contracts. Since exerting effort is costly and reaps benefits only to the principal, in and by itself the agent has no incentive to exert effort. This challenge is often referred to as "moral hazard". To incentivize the agent to exert effort, the principal specifies a contract that maps project outcomes (in this case, "success" and "failure") to payments made to the agent by the principal. It is well known that in the binary-outcome case, it is without loss of generality to assume that the payment for failure is 0. Thus, a contract can be fully described by a parameter $\alpha \in [0, 1]$, which is the fraction of the principal's reward that is paid to the agent (in our case, where the reward is normalized to 1, α is essentially the payment for success).

Under a contract $\alpha \in [0,1]$, the agent's utility from action $i \in A$ is the expected payment minus the cost, i.e.,

$$u_A(\alpha, i) = p_i \cdot \alpha - c_i.$$

The principal's utility under a contract α and an agent's action i is the expected reward minus the expected payment, i.e.,

$$u_P(\alpha, i) = p_i(1 - \alpha).$$

Given a contract α, the agent's best response is an action that maximizes his utility, namely $i_\alpha \in \arg\max_{i \in A} u_A(\alpha, i)$. As standard in the literature, the agent breaks ties in favor of the principal's utility. The principal's problem, our problem in this paper, is to find the contract α that maximizes her utility, given that the agent best responds. Let $u_P(\alpha) = u_P(\alpha, i_\alpha)$, then the principal's objective is to find α that maximizes $u_P(\alpha)$.

2.2 Combinatorial Contract Settings

In what follows, we define two combinatorial settings, one with multiple agent (introduced by [2], as presented in [9]), the other with multiple actions (introduced by [8]). In both cases we use a set function $f : 2^A \to [0, 1]$ that maps every subset of a set A to a success probability. The set A denotes the set of agents in the first setting, and the set of actions in the second setting. When considering f outside of a specific context, we refer to A as the set of items.

We focus on success probability functions f that belong to one of the following classes of complement-free set functions [22]. A set function $f : 2^A \to \mathbb{R}_{\geq 0}$ is:

1. *Additive* if there exist values $v_1, \ldots, v_n \in \mathbb{R}_{\geq 0}$ such that $\forall S \subseteq A.\ f(S) = \sum_{i \in S} v_i$.
2. *Coverage* if there is a set of elements U, with associated positive weights $\{w_u\}_{u \in U}$, and a mapping $h : A \to 2^U$ such that for every $S \subseteq A$, $f(S) = \sum_{u \in U} w_u \cdot 1[\exists i \in S.\ u \in h(i)]$, where $1[B]$ is the indicator variable of the event B. In this paper, we focus on a special case of coverage functions, called *normalized unweighted coverage functions*, in which $w_u = \frac{1}{|U|}$ for every $u \in |U|$. We represent a normalized unweighted coverage function f using a tuple (U, A, h), for which $f(S) = \frac{1}{|U|} \left|\bigcup_{i \in S} h(i)\right|$.
3. *Submodular* if for any two sets $S, S' \subseteq A$ s.t. $S \subseteq S'$ and $i \in A$ it holds that $f(i \mid S) \geq f(i \mid S')$, where $f(i \mid S) = f(S \cup \{i\}) - f(S)$ is the marginal contribution of i to S.
4. *XOS* if there exists a finite collection of additive functions $\{a_i : 2^A \to \mathbb{R}_{\geq 0}\}_{i=1}^k$ such that for every $S \subseteq A$, $f(S) = \max_{i=1,\ldots,k} a_i(S)$.

It is well known that additive $\subset$ coverage $\subset$ submodular $\subset$ XOS, and all containment relations are strict [20].

Computational model. Since the success probability function $f : 2^A \to [0, 1]$ contains exponentially many (in $|A|$) values, we assume, as is common in the literature, that the algorithm has a value oracle access, which, for every set $S \subseteq A$, returns $f(S)$. It should be noted that most of our results hold under an even stronger assumption. Namely, that the success probability function f admits a succinct representation, for which a value oracle can be computed efficiently, and that this representation is given to the algorithm. This assumption implies that these results are purely computational hardness ones, as the algorithm essentially knows the entire function.

We next present the two combinatorial models considered in this paper.

Setting 1: Multiple agents. In the multiple agents setting, the principal interacts with a set A of n agents. Every agent $i \in A$ decides whether to exert effort or not (binary action). Exerting effort comes with a cost of $c_i \geq 0$ (otherwise the cost is zero). The success probability function $f : 2^A \to [0, 1]$ maps every set of agents who exert effort to a success probability of the project, where $f(S)$ denotes the success probability if S is the set of agents who exert effort.

A contract is now a vector $\alpha = (\alpha_1, \ldots, \alpha_n) \in [0, 1]^n$, where α_i is the payment to agent i upon a project success.

Given a contract $\alpha = (\alpha_1, \ldots, \alpha_n)$ and a set S of agents who exert effort, the principal's utility is given by $\left(1 - \sum_{i \in A} \alpha_i\right) f(S)$. Agent i's utility is given by $\alpha_i f(S) - 1[i \in S]c_i$. Note that agent i is paid in expectation $\alpha_i f(S)$ regardless of whether he exerts effort or not, but pays c_i only if he exerts effort (i.e., if $i \in S$).

To analyze contracts, we consider the (pure) Nash equilibria of the induced game among the agents. A contract $\alpha = (\alpha_1, \ldots, \alpha_n)$ is said to incentivize a set $S \subseteq A$ of agents to exert effort (in equilibrium) if

$$\begin{aligned} &\alpha_i f(S) - c_i \geq \alpha_i f(S \setminus \{i\}) && \text{for all } i \in S, \text{ and} \\ &\alpha_i f(S) \geq \alpha_i f(S \cup \{i\}) - c_i && \text{for all } i \notin S. \end{aligned}$$

Since equilibria may not be unique, we think of a contract as a pair of α, S where S is a set of agents incentivized to exert effort (in equilibrium).

It is easy to observe that for any set $S \subseteq A$, the best way for the principal to incentivize the agents in S is by the contract

$$\begin{aligned} \alpha_i &= \frac{c_i}{f(i \mid S \setminus \{i\})} && \text{for all } i \in S, \text{ and} \\ \alpha_i &= 0 && \text{for all } i \notin S, \end{aligned}$$

where $f(i \mid S \setminus \{i\}) = f(S) - f(S \setminus \{i\})$ is the marginal contribution to S of adding i to $S \setminus \{i\}$. We interpret $\frac{c_i}{f(i|S\setminus\{i\})}$ as 0 if $c_i = 0$ and $f(i \mid S \setminus \{i\}) = 0$ and as infinity when $c_i > 0$ and $f(i \mid S \setminus \{i\}) = 0$. The principal thus tries to find a set S that maximizes $g(S)$ where

$$g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f(i \mid S \setminus \{i\})}\right) f(S).$$

Let $S^\star$ be the optimal set of agents, i.e., the set that maximizes g. We say that S is a β-approximation to the optimal contract (where $\beta \leq 1$) if $g(S) \geq \beta \cdot g(S^\star)$.

Setting 2: Multiple actions. In the multiple actions setting, the principal interacts with a single agent, who faces a set A of n actions, and can choose any subset $S \subseteq A$ of them. Every action $i \in A$ is associated with a cost $c_i \geq 0$, and the cost of a set S of actions is $\sum_{i \in S} c_i$. The success probability function $f(S)$ denotes the probability of a project success when the agent chooses the set of actions S.

A contract is defined by a single parameter $\alpha \in (0, 1)$, which denotes the payment to the agent upon the project's success. Given a contract α, the agent's and principal's utilities under a set of actions S are, respectively,

$$u_A(\alpha, S) = f(S) \cdot \alpha - \sum_{i \in S} c_i \qquad \text{and} \qquad u_P(\alpha, S) = f(S)(1 - \alpha).$$

The agent's best response for a contract α is $S_\alpha \in \arg\max_{S\subseteq A} u_A(\alpha, S)$. As before, the agent breaks ties in favor of the principal's utility. We also denote $u_P(\alpha) = u_P(\alpha, S_\alpha)$, and the principal's objective is to find a contract α that maximizes her utility $u_P(\alpha)$. We denote by $\alpha^\star$ the optimal contract, i.e., the contract that maximizes $u_P(\alpha^\star)$. We say that a contract α is a β-approximation (where $\beta \leq 1$) if $u_P(\alpha) \geq \beta \cdot u_P(\alpha^\star)$.

3 An NP-hard Promise Problem of Coverage Functions

In this section we define a promise problem regarding normalized unweighted coverage functions that is the basis of our hardness results for the contract models with submodular success probability functions. Essentially, we show that it is NP-hard to distinguish between a normalized unweighted coverage function f that has a relatively small set S with $f(S) = 1$, and one for which a significantly larger set T would be required to get close to $f(T) = 1$. This naturally leads to our hardness results in Sections 4 and 6; by setting uniform costs to all actions / agents, an approximately optimal contract can usually distinguish between the case where a relatively small set achieves $f(S) = 1$ (and incentivizing costs less to the principal) and a larger set is required to get close to $f(T) = 1$ (and incentivizing costs more to the principal).

This hardness result is an extension of a hardness result by Feige [13], which we present next for completeness. Recall that a normalized unweighted coverage function f is given by a tuple (U, A, h), where U is a set of elements, and h is a mapping from A to 2^U (see Section 2.2).

▶ **Proposition 1** ([13]). *For every $0 < \varepsilon < e^{-1}$, on input (k, f), where $k \in \mathbb{N}$ and $f = (U, A, h)$ is a normalized unweighted coverage function such that exactly one of the following two conditions holds:*

1. *There exists a set $S \subseteq A$ of size k such that $f(S) = 1$.*
2. *Every set $S \subseteq A$ of size k satisfies $f(S) \leq 1 - e^{-1} + \varepsilon$.*

It is NP-hard to determine which of the two conditions is satisfied by the input.

Remark: Feige [13] used a different terminology, but proved an equivalent result.

We next present the following extension to Proposition 1.

▶ **Proposition 2.** *For every $M > 1$ and every $0 < \varepsilon < e^{-1}$, on input (k, f), where $k \in \mathbb{N}$ and $f = (U, A, h)$ is a normalized unweighted coverage function such that $\forall i \in A.\ f(\{i\}) = \frac{1}{k}$ and exactly one of the following two conditions holds:*

1. *There exists a set $S \subseteq A$ of size k such that $f(S) = 1$.*
2. *Every set $S \subseteq A$ of size βk such that $\beta \leq M$ satisfies $f(S) \leq 1 - e^{-\beta} + \varepsilon$.*

It is NP-hard to determine which of the two conditions is satisfied by the input.

Proposition 1 asserts that it is hard to approximate the maximum value of $f(S)$ for sets of size k within some factor, and in Proposition 2 we generalize this to the maximum value of $f(S)$ over all sets S of some fixed size $\ell \in O(k)$. This is a necessary adjustment for our contract design problem, as we are not restricted to sets of size exactly k. Indeed, we can have either smaller or slightly larger sets (at smaller or slightly larger costs, respectively).

The proof of Proposition 2 can be found in the full version of this paper.

4 Hardness of Approximation for Multi-Agent, Submodular f

In this section, we settle an open problem from [9]. In particular, [9] show that in a multi-agent setting, one can get constant-factor approximation for settings with submodular success probability function f, with value queries. It is left open whether one can get better than constant approximation for this setting. The following theorem resolves this question in the negative.

▶ **Theorem 3.** *In the multi-agent model, for submodular (and even normalized unweighted coverage) success probability function f, no polynomial time algorithm with value oracle access can approximate the optimal contract within a factor of* 0.7*, unless $P = NP$.*

Before presenting the proof of this theorem, we recall some of the details of the multi-agent model (see Section 2). In this setting, the principal interacts with a set A of n agents. Every agent $i \in A$ decides whether to exert effort (at cost $c_i \geq 0$) or not. The success probability function $f : 2^A \to [0, 1]$ maps every set of agents who exert effort to a success probability of the project. A contract is a vector $\alpha = (\alpha_1, \dots, \alpha_n)$ specifying the payment to each agent upon success. The principal seeks to find the optimal set S of agents to exert effort, which is equivalent to maximizing the function

$$g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f(i \mid S \setminus \{i\})}\right) f(S).$$

Proof of Theorem 3. Our proof relies on Proposition 2, by creating a reduction with the following properties: Given as input (k, f) where $f : 2^A \to [0, 1]$ is a coverage function that satisfies one of the conditions of Proposition 2, we construct an instance of the multi-agent contract problem with the following separation: Under condition (1) in the proposition, the principal's utility is at least 0.5, whereas under condition (2), the principal's utility is strictly less than 0.35.

Suppose we have a 0.7-approximation algorithm for our contract problem, and let (α, S) be the output (i.e., contract) of this algorithm on our reduction-generated instance. If the principal's utility under (α, S) is greater than or equal to 0.35 (note that computing the principal's utility is easy in this model), then we must be under condition (1) of Proposition 2. Otherwise (if the principal's utility is less than 0.35), then we must be under condition (2), since for condition (1) to hold, we should get at least $0.7 \cdot 0.5 = 0.35$. Since the construction of our reduction is polynomial, this proves the theorem.

It remains to construct an instance that admits the separation above.

Given $k, f : 2^A \to [0, 1]$ from Proposition 2 with $M = 2, \varepsilon = 0.01$, we construct an instance of the multi-agent contract problem, where A is the set of agents, f is the success probability function, and $c_i = \frac{1}{2k^2}$ for every agent $i \in A$. We next establish the desired separation.

Case 1: f satisfies condition (1) from Proposition 2. Take set S per condition (1) of Proposition 2 (i.e., $|S| = k$ and $f(S) = 1$). We claim that for any $i \in S$, $f(i \mid S \setminus \{i\}) = \frac{1}{k}$. Indeed,

$$f(i \mid S \setminus \{i\}) = f(S) - f(S \setminus \{i\}) \geq 1 - (k-1)\frac{1}{k} = \frac{1}{k},$$

where the inequality follows by $f(S) = 1$ and submodularity of f. In the other direction we have $f(i \mid S \setminus \{i\}) \leq f(\{i\}) = \frac{1}{k}$, giving $f(i \mid S \setminus \{i\}) = \frac{1}{k}$. We now have

$$g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f(i \mid S \setminus \{i\})}\right) f(S) = \left(1 - \sum_{i \in S} k \cdot c_i\right) f(S) = \left(1 - \frac{|S|}{2k}\right) \cdot 1 = \frac{1}{2}.$$

We get that the principal's utility under the optimal contract is at least 0.5, as desired. [1]

Case 2: f satisfies condition (2) from Proposition 2. Let $S \subseteq A$ be an arbitrary set. We show that $g(S) < 0.35$.

If $|S| \geq 2k$, we have

$$g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f(i \mid S \setminus \{i\})}\right) f(S) \leq \left(1 - \sum_{i \in S} \frac{c_i}{f(\{i\})}\right) f(S) = \left(1 - \frac{|S|}{2k}\right) f(S) \leq 0,$$

where the first inequality is by submodularity of f, and the last inequality is by $|S| \geq 2k$.

If $|S| < 2k = Mk$, we can apply condition (2) and get

$$g(S) \leq \left(1 - \sum_{i \in S} \frac{c_i}{f(\{i\})}\right) f(S) \leq \left(1 - \frac{|S|}{2k}\right) \left(1 - e^{-\frac{|S|}{k}} + 0.01\right) < 0.35,$$

where the third inequality is since $(1 - \frac{x}{2})(1 - e^{-x} + 0.01) < 0.35$ for all x. This concludes the proof. ◀

In the full version of this paper, we discuss the differences in our approach for hardness results, and the approach of [9], and explain why the hardness results in Dütting et al. [9] cannot be extended to show hardness of getting a PTAS for submodular success probability functions.

5 Hardness of Approximation for Multi-Agent, XOS f

In this section, we show that, in the multi-agent model, one cannot approximate the optimal contract under XOS success probability functions within a constant factor, with access to a value oracle. More formally, we prove the following theorem:

▶ **Theorem 4.** *In the multi-agent model, with XOS success probability functions, no (randomized) algorithm that makes poly-many value queries can approximate the optimal contract (with high probability) to within a factor greater than $4n^{-1/6}$.*

To prove this theorem, we define a probability distribution over XOS success probability functions and show an upper bound on the expected performance of any deterministic algorithm. By Yao's principle, this gives us an upper bound on the worst-case performance of a randomized algorithm, thus proving the theorem.

We note that unlike our other results, which are computational hardness results, this result is information theoretical, as we rely on the algorithm "not knowing" something about the success probability function.

[1] We note that in this case, incentivizing S is the optimal contract, since to incentivize each agent the principal needs to pay at least $\frac{1}{2k}$, and since the success probability function satisfies, for any $T \subseteq A$, $f(T) \leq \frac{|T|}{k}$, we get that $g(T) \leq (1 - \frac{|T|}{2k})\frac{|T|}{k} \leq \frac{1}{2}$.

Proof. For any n, we uniformly at random sample a subset $G \subseteq A = [n]$ of the agents of size $|G| = m = n^{1/3}$, and build the following instance:

1. Our success probability function is $f_G : 2^A \to [0,1]$, and for any non-empty set $S \subseteq A$:

$$f_G(S) = \frac{1}{n} \max\left(|S \cap G|, \sqrt{m}, \frac{|S|}{\sqrt{m}}\right).$$

2. For any $i \in A$ the cost associated with agent i is $c_i = \frac{1}{2m \cdot n}$.

Note that f_G is clearly XOS, as it is the maximum of three XOS functions (two of them are additive and one of them is unit demand), and the family of XOS is closed under maximum (in contrast to submodular).

We start by noting that

$$g(G) = \left(1 - \sum_{i \in G} \frac{c_i}{f_G(i \mid G \setminus \{i\})}\right) f_G(G) = \left(1 - m \cdot \frac{n}{2m \cdot n}\right) \cdot \frac{m}{n} = \frac{m}{2n}.$$

We call a value query on set S *successful* if $|S| \leq m^{1.5}$ and $|S \cap G| > \sqrt{m}$, and *unsuccessful* otherwise.

We start by noting that the probability of any specific value query being successful is negligible:

▶ **Lemma 5.** *Let $S \subseteq A$. For $n \geq 512$, it holds that*

$$Pr_G[S \textit{ is successful}] \leq e^{-\frac{\sqrt{m}}{4}}.$$

Proof. The probability of S being successful when G is chosen uniformly at random from all subsets of A of size m is

$$\Pr_G[|S \cap G| > \sqrt{m} \wedge |S| \leq m^{1.5}]$$

which is monotone in the size of S up to size $m^{1.5}$. For $|S| = m^{1.5}$, $|S \cap G|$ is distributed as a hyper-geometric random variable $HG(n, m^{1.5}, m)$. Therefore

$$\begin{aligned} Pr_G[S \text{ is successful}] &\leq Pr_{X \sim HG(n,m^{1.5},m)}[X > \sqrt{m}] \leq Pr_{X \sim Bin(m, \frac{m^{1.5}}{n-m})}[X > \sqrt{m}] \\ &\leq Pr_{X \sim Bin(m, \frac{2m^{1.5}}{n})}[X > \sqrt{m}], \end{aligned}$$

where the second inequality is because the probability of success in each one of the m draws (without replacement) in the hyper-geometric distribution $HG(n, m^{1.5}, m)$ is always at most $\frac{m^{1.5}}{n-m}$, and the last inequality is because for $n \geq 512$ it holds that $\frac{m^{1.5}}{n-m} \leq \frac{2m^{1.5}}{n}$.

When denoting $\mu = E_{X \sim Bin(m, \frac{2m^{1.5}}{n})}[X] = \frac{2m^{2.5}}{n} = \frac{2}{\sqrt{m}}$ and $\delta = \frac{\sqrt{m}}{2\mu} = \frac{m}{4}$ we can apply the Chernoff bound and get

$$\begin{aligned} \Pr_{X \sim Bin(m, \frac{2m^{1.5}}{n})}[X > \sqrt{m}] &\leq \Pr_{X \sim Bin(m, \frac{2m^{1.5}}{n})}[X \geq (1+\delta)\mu] \\ &\leq e^{\frac{-\delta^2 \mu}{2+\delta}} \leq e^{\frac{-\delta^2 \mu}{2\delta}} = e^{-\frac{m^{1.5}/8}{m/2}} = e^{-\frac{\sqrt{m}}{4}}. \end{aligned}$$ ◀

We now note that any set $S \subseteq A$ that is an unsuccessful value query is also a poor approximation of the optimal contract:

▶ **Lemma 6.** *Let $S \subseteq A$. For $n \geq 64$, if a value query on S is unsuccessful, then $g(S) \leq \frac{4g(G)}{\sqrt{m}}$.*

Proof. We start by noting that for any $S \subseteq A$ and $i \in S$, it holds that $f_G(i \mid S \setminus \{i\}) \leq \frac{1}{n}$, which implies $\frac{c_i}{f_G(i|S\setminus\{i\})} = \frac{1/(2mn)}{f_G(i|S\setminus\{i\})} \geq \frac{1}{2m}$.

Now, if $|S| \geq 2m$, then $g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f_G(i|S\setminus\{i\})}\right) f(S) \leq \left(1 - |S| \cdot \frac{1}{2m}\right) f(S) \leq 0$, as needed.

Otherwise, since $n \geq 64$, we have $|S| < 2m \leq m^{\frac{3}{2}}$, and by our definition of an unsuccessful value query, it holds that $|S \cap G| \leq \sqrt{m}$, meaning $g(S) \leq f_G(S) \leq \frac{2\sqrt{m}}{n} = \frac{4g(G)}{\sqrt{m}}$, as needed. ◀

From Lemma 6, if we assume without loss of generality that an algorithm value queries the set S which it returns as a contract, we can say that an algorithm with no successful value queries achieves at most a $\frac{4}{\sqrt{m}}$-approximation. We therefore conclude the proof by showing that the probability of any algorithm that makes a polynomial number of value queries having a successful query is negligible.

▶ **Lemma 7.** *For $n \geq 512$, if a determinstic algorithm makes at most k value queries, the probability of at least one query being successful is at most $k \cdot e^{-\frac{\sqrt{m}}{4}}$.*

Proof. For any m such that $k \cdot e^{-\frac{\sqrt{m}}{4}} \geq 1$ we are done. Otherwise, for m such that $k \cdot e^{-\frac{\sqrt{m}}{4}} < 1$, let ALG be a deterministic algorithm that makes at most k value queries. Our first step is to show that adaptivity doesn't help ALG, which allows us to apply union bound. More precisely, we show the existence of non-adaptive queries $S_1, \dots, S_\ell$ such that $\ell \leq k$ and

$$\Pr_G[ALG \text{ makes a successful query}] \leq \Pr_G[\text{one of } S_1, \dots, S_\ell \text{ is a successful query}].$$

Let S_i be the i-th query that ALG asks after the answers to all previous queries S_j were $\frac{1}{n} \max\left(\sqrt{m}, \frac{|S_j|}{\sqrt{m}}\right)$ for all $j < i$, and let $\ell \leq k$ be the index of the last query asked in this scenario[2]. To see that

$$\Pr_G[ALG \text{ makes a successful query}] \leq \Pr_G[\text{one of } S_1, \dots, S_\ell \text{ is a successful query}],$$

let $T_1, \dots, T_{\ell'}$ be the (perhaps adaptive) queries ALG actually makes. We will show that if at least one of those is successful, then at least one of $S_1, \dots, S_\ell$ is successful. Assume that one of $T_1, \dots, T_{\ell'}$ is successful, and let i be the lowest index such that T_i is successful. If $i = 1$, note that since ALG is deterministic, then $S_1 = T_1$, as needed. Otherwise, by definition of i, for any $j < i$, T_j is an unsuccessful query, then the answer of ALG to value query T_j is $\frac{1}{n} \max\left(\sqrt{m}, \frac{|T_j|}{\sqrt{m}}\right)$, which by induction gives us for any $j \leq i$, $T_j = S_j$, meaning S_i is a successful query, as needed.

Now, by the union bound and Lemma 5 we have

$$\begin{aligned} Pr_G[ALG \text{ makes a successful query}] &\leq Pr_G[\text{one of } S_1, \dots, S_\ell \text{ is a successful query}] \\ &\leq k \cdot e^{-\frac{\sqrt{m}}{4}}. \end{aligned}$$ ◀

Note that for any k that is polynomial in n it holds that $k \cdot e^{-\frac{\sqrt{m}}{4}} = k \cdot e^{-\frac{n^{1/6}}{4}}$ is negligible, which concludes the proof. ◀

[2] Since $k \cdot e^{-\frac{\sqrt{m}}{4}} < 1$, the sequence of sets $S_1, \dots, S_\ell$ is well defined since by union bound and Lemma 5 there is a positive probability that all of $S_1, \dots, S_\ell$ are unsuccessful, in which case the answer to query S_j is indeed $\frac{1}{n} \max\left(\sqrt{m}, \frac{|S_j|}{\sqrt{m}}\right)$.

6 Hardness of Approximation for Multi-Action, Submodular f

In this section, we use Proposition 2 to strengthen a hardness result of [8], showing that the optimal contract for multi-action settings with submodular f is not only hard to compute exactly, but also to approximate within any constant.

Before presenting our result, we recall the multi-action model from Section 2. The principal interacts with a single agent, who faces a set A of n actions, and can choose any subset $S \subseteq A$ of them. Every action $i \in A$ is associated with a cost $c_i \geq 0$, and the cost of a set S of actions is $\sum_{i \in S} c_i$. The success probability function $f(S)$ denotes the probability of a project's success when the agent chooses the set of actions S. A contract is defined by a single parameter $\alpha \in [0,1]$, which denotes the payment to the agent upon the project's success. The principal's objective is to find a contract α that maximizes her utility $u_P(\alpha) = (1-\alpha)f(S_\alpha)$, where $S_\alpha \in \arg\max_{S \subseteq A} u_A(\alpha, S) = \arg\max_{S \subseteq A} f(S) \cdot \alpha - \sum_{i \in S} c_i$ is the agent's best response to a contract α, with tie-breaking in favor of the principal.

We are now ready to present the theorem.

▶ **Theorem 8.** *In the multi-action model, for submodular (and even normalized unweighted coverage) success probability function f, no polynomial time algorithm with value oracle access can approximate the optimal contract within any constant factor, unless $P = NP$.*

Proof. Let $\beta \in \left(0, \frac{1}{12}\right)$, we prove that no poly-time 12β-approximation algorithm exists, by reducing from the hardness presented in Proposition 2.

Let $(k, f' = (U', A', h'))$ be the input to our reduction per Proposition 2 with $M = 2, \varepsilon = \beta^4$. We build the following contract instance:

- The set of actions is $A = A' \cup \{0\}$.
- The success probability function is $f(S) = \frac{1}{2}(f'(S \cap A') + 1[0 \in S])$.
- The costs are $c_i = \frac{1-\beta^2}{2k}$ for all $i \in A'$ and $c_0 = \frac{1}{2}(1-\beta^3)$.

▶ **Lemma 9.** *The success probability function defined above is a normalized unweighted coverage function.*

Proof. Let $U = U' \times \{0,1\}$, and define

$$h(i) = \begin{cases} h'(i) \times \{0\} & i \in A' \\ U' \times \{1\} & i = 0 \end{cases}$$

The normalized unweighted coverage function defined by (U, A, h) is equal to f. ◀

▶ **Lemma 10.** *In the contract problem instance defined above, the following holds:*

1. *If (k, f') satisfies condition 1 of Proposition 2, any α_0 which is a 12β-approximation of the optimal contract satisfies $\alpha_0 < 1 - \beta^3$.*
2. *If (k, f') satisfies condition 2 of Proposition 2, any α_0 which is a 12β-approximation of the optimal contract satisfies $\alpha_0 \geq 1 - \beta^3$.*

Note that by proving Lemma 10 we conclude the proof of Theorem 8.

Proof of Lemma 10. We first observe that the agent's best response to a contract $\alpha < 1-\beta^2$ is to take no actions. This holds since for any $i \in A'$ it holds that

$$\alpha f(\{i\}) - c_i = \alpha \cdot \frac{1}{2} f'(\{i\}) - c_i = \alpha \cdot \frac{1}{2k} - \frac{1-\beta^2}{2k} < 0,$$

where the second equality is since Proposition 2 guarantees $f'(\{i\}) = \frac{1}{k}$ for any $i \in A'$. It also holds that for $i = 0$ we have

$$\alpha f(\{i\}) - c_i = \alpha \cdot \frac{1}{2} - \frac{1}{2}(1-\beta^3) < 0.$$

This means the agent's utility from any non-empty set $S \subseteq A$ is strictly less than 0, since

$$u_A(\alpha, S) = \alpha f(S) - \sum_{i \in S} c_i \leq \alpha \sum_{i \in S} f(\{i\}) - \sum_{i \in S} c_i = \sum_{i \in S} \alpha f(\{i\}) - c_i < 0,$$

where the first inequality follows by subadditivity.

Note that the same arguments show that, given the contract $\alpha = 1 - \beta^2$, the agent has non-positive utility from any non-empty set $S \subseteq A$.

Case 1: (k, f') satisfies the first condition of Proposition 2. Let $S \subseteq A'$ be a set that satisfies the condition (i.e. $|S| = k$ and $f'(S) = 1$). Under the contract $\alpha = 1 - \beta^2$, the agent's utility from the set S is

$$u_A(\alpha, S) = \alpha f(S) - \sum_{i \in S} c_i = (1-\beta^2)\frac{1}{2} - |S|\frac{1-\beta^2}{2k} = 0.$$

Since, as we noted earlier, no set has a greater utility to the agent, and ties are broken in favor of the principal, this implies that the agent's best response S_α satisfies $f(S_\alpha) \geq f(S) \geq \frac{1}{2}$. It follows that the principal's utility from the contract $\alpha = 1 - \beta^2$ satisfies

$$u_P(\alpha) = f(S_\alpha)(1-\alpha) \geq \frac{1}{2}\beta^2.$$

Now, let α_0 be some 12β-approximation of the optimal contract. This implies that

$$u_P(\alpha_0) \geq 12\beta \cdot u_P(1-\beta^2) \geq 12\beta \cdot \frac{1}{2}\beta^2 > \beta^3.$$

On the other hand, trivially it holds that $u_P(\alpha_0) \leq 1 - \alpha_0$, which gives us $\alpha_0 < 1 - \beta^3$, as needed.

Case 2: (k, f') satisfies the second condition of Proposition 3.2. We start by arguing that the principal's utility from the contract $\alpha = 1 - \beta^3$ is at least $\frac{1}{2}\beta^3$. First, we show that the agent's best response S_α will always include the action 0. Indeed, if we assume by contradiction that $0 \notin S_\alpha$, by adding 0 to S_α we do not change the agent's utility:

$$u_A(S_\alpha \cup \{0\}) - u_A(S_\alpha) = \alpha f(0 \mid S_\alpha) - c_0 = (1-\beta^3) \cdot \frac{1}{2} - \frac{1}{2}(1-\beta^3) = 0.$$

This means that $S_\alpha \cup \{0\}$ has the same utility to the agent, but a greater utility to the principal, contradicting tie-breaking in favor of the principal. This implies that

$$u_P(\alpha) = f(S_\alpha)(1-\alpha) \geq \frac{1}{2}(1-\alpha) = \frac{1}{2}\beta^3.$$

Now, let $\alpha < 1 - \beta^3$ be some contract, we show that $u_P(\alpha) < 6\beta^4 \leq 12\beta \cdot u_P(1-\beta^3)$, completing the proof of Lemma 10. If $\alpha < 1 - \beta^2$, as argued before, the agent's best response is the empty set, which means the principal's utility is 0 and we are done. Otherwise, if $1 - \beta^2 \leq \alpha < 1 - \beta^3$, since S_α is the agent's best response, it is clear that $0 \notin S_\alpha$ (since otherwise $S_\alpha \setminus \{0\}$ has a strictly better utility to the agent). From this we conclude that $f(S_\alpha) = \frac{1}{2}f'(S_\alpha) \leq \frac{1}{2}$. Since S_α must have a non-negative utility to the agent, it also holds that

$$|S_\alpha|\frac{1-\beta^2}{2k} = \sum_{i \in S_\alpha} c_i \leq \alpha f(S_\alpha) \leq f(S_\alpha). \tag{1}$$

Since $f(S_\alpha) \leq \frac{1}{2}$, Inequality 1 implies that $|S_\alpha| \leq \frac{k}{1-\beta^2} \leq M \cdot k$. This allows us to use condition (2) of Proposition 2, which implies that $f(S_\alpha) \leq \frac{1}{2}(1 - e^{-|S_\alpha|/k} + \varepsilon)$. Denoting $x = \frac{|S_\alpha|}{k}$, and plugging this into Inequality 1, we get the inequality

$$x\frac{1-\beta^2}{2} \leq \frac{1}{2}(1-e^{-x}+\varepsilon) \leq \frac{1}{2}\left(x - \frac{1}{4}x^2 + \varepsilon\right),$$

where the last inequality is since $e^{-x} \geq 1 - x + \frac{1}{4}x^2$ for any $x \in [0,2]$. By rearranging, we get that

$$\frac{1}{4}x^2 - \beta^2 x - \varepsilon \leq 0,$$

implying

$$x \leq \frac{\beta^2 + \sqrt{\beta^4+\varepsilon}}{1/2} \leq \frac{\beta^2 + \sqrt{2}\beta^2}{1/2} < 6\beta^2$$

This means the principal's utility is at most

$$u_P(\alpha) = f(S_\alpha)(1-\alpha) = \frac{1}{2}f'(S_\alpha)(1-\alpha) \leq |S_\alpha| \cdot \frac{1}{2k}(1-\alpha) < 6\beta^2(1-\alpha) < 6\beta^4,$$

as needed. ◀

This concludes the proof of Theorem 8. ◀

7 Hardness of Approximation for Multi-Action, XOS f

In this section, we show a hardness of approximation result for the multi-action model with XOS success probability functions. More formally, we prove the following theorem:

▶ **Theorem 11.** *In the multi-action model, for XOS f, for any $\varepsilon > 0$, no polynomial time algorithm with value query access can approximate the optimal contract to within a factor of $n^{-\frac{1}{2}+\varepsilon}$, unless P=NP.*

Our proof of Theorem 11 relies on the hardness of approximating $\omega(G)$, which is the size of the largest clique in the graph G. This hardness result was presented by [16, 25], who prove the following theorem:

▶ **Theorem 12** ([16, 25])**.** *For all $\varepsilon > 0$, it is NP-hard to approximate $\omega(G)$ to within $n^{-1+\varepsilon}$.*

Our technique is to use any β-approximation algorithm of the optimal contract to distinguish between the cases $\omega(G) \leq \delta$ and $\omega(G) \geq \frac{2\delta}{\beta^2}$, for any $\delta > 0$. Solving this promise problem allows us to get a guarantee of either $\omega(G) > \delta$ or $\omega(G) < \frac{2\delta}{\beta^2}$. By iterating over $\delta_i = 2^i$, we can get some i for which we are guaranteed $\omega(G) > \delta_i$ and $\omega(G) < \frac{2\delta_{i+1}}{\beta^2}$. This allows us to approximate $\omega(G)$ within a factor of $\frac{\beta^2}{4}$ (see formal arguments in Lemma 17), which implies Theorem 11 from [16, 25].

In Section 7.1 we show how to use a β-approximation algorithm for the contract problem to distinguish between the two cases $\omega(G) \leq \delta$ and $\omega(G) \geq \frac{2\delta}{\beta^2}$ in polynomial time for any $\delta > 0$, and in Section 7.2 we formally prove that this distinction gives us the ability to approximate $\omega(G)$ to within a factor of $\frac{\beta^2}{4}$ in polynomial time, thus concluding the proof of Theorem 11.

7.1 Distinguishing Between $\omega(G) \leq \delta$ and $\omega(G) \geq \frac{2\delta}{\beta^2}$

In this section, we prove the following lemma:

▶ **Lemma 13.** *Algorithm 1 runs in polynomial time, given oracle access to a β-approximation of the optimal contract for XOS functions. Additionally, on input (G, δ) composed of a graph G and a positive integer δ it holds that:*

1. *If $\omega(G) \leq \delta$, then Algorithm 1 returns SMALL.*
2. *If $\omega(G) \geq \frac{2\delta}{\beta^2}$, then Algorithm 1 returns LARGE.*
3. *If $\delta < \omega(G) < \frac{2\delta}{\beta^2}$, then Algorithm 1 returns either SMALL or LARGE.*

We note that in Algorithm 1 we build a contract problem instance with a success probability function that attains values greater than 1. This is done for simplicity, and the result clearly holds for success probability functions that attain values within $[0, 1]$, by normalizing both f and the costs c_i by $f(V')$ (the maximum value of f).

Algorithm 1 Distinguishing Between $\omega(G) \leq \delta$ and $\omega(G) \geq \frac{2\delta}{\beta^2}$.

1: Given a graph $G = (V, E)$ and $\delta \in \mathbb{N}^+$, build the graph $G' = (V', E')$ such that

$$V' = V \cup [\delta]$$
$$E' = E \cup \{\{i, j\} \mid i, j \in [\delta] \wedge i \neq j\}$$

2: Denote $\varepsilon = \frac{2}{\beta} - 1$, $M = |V'| + \varepsilon$

3: Get α_0 which is a β-approximation to the optimal contract in the instance $(V', f : 2^{V'} \to \mathbb{R}_{\geq 0}, \{c_i = M\}_{i \in V'})$ where

$$f(S) = (M + 1[S \text{ is a clique in } G']) \cdot |S| + \min(|S|, \delta) \cdot \varepsilon$$

4: If $\alpha_0 < \frac{M}{M+1}$, return SMALL, otherwise return LARGE.

First, we note the following lemma, which proves our oracle call in step 2 is valid:

▶ **Lemma 14.** *The function f as defined in step 3 of Algorithm 1 is monotone, XOS, and value queries can be computed in polynomial time.*

Proof. For any non-empty subset $T \subseteq V'$, we define the additive function $f_T : V' \to \mathbb{R}_{\geq 0}$ according to the weights

$$a_i^T = \begin{cases} M + 1[T \text{ is a clique}] + \varepsilon \cdot \frac{\min(|T|, \delta)}{|T|} & i \in T \\ 0 & else, \end{cases}$$

where $f_T(S) = \sum_{i \in S} a_i^T$.

Let $S \subseteq V'$, we claim that

$$f(S) = \max_{T \in 2^{V'} \setminus \{\emptyset\}} f_T(S),$$

which proves both XOS and monotonicity since all weights a_i^T are non-negative. First, by taking $T = S$, it is clear that

$$\max_{T\in 2^{V'}\setminus\{\emptyset\}} f_T(S) \geq f_S(S) = f(S).$$

Let $T \subseteq V'$. We will show that $f_T(S) \leq f(S)$, thus concluding the proof.

If $S \subseteq T$, it is clear that for any $i \in S$ we have $a_i^T \leq a_i^S$ which implies $f_T(S) \leq f_S(S) = f(S)$.

Otherwise, if $S \not\subseteq T$, it is clear that $|S \cap T| < |S|$, which gives us

$$\begin{aligned} f_T(S) = \sum_{i\in S} a_i^T &\leq |S\cap T|\left(M + 1 + \varepsilon\frac{\min(|T|,\delta)}{|T|}\right) \leq |S\cap T|(M+1) + \varepsilon\min(|S|,\delta) \\ &< (|S\cap T|+1)M + \varepsilon\min(|S|,\delta) \leq |S|\cdot M + \varepsilon\min(|S|,\delta) \leq f(S), \end{aligned}$$

where the strict inequality is since $M > |S \cap T|$.

A value oracle of f can be computed efficiently by checking, for a given set S, whether it is a clique or not. ◀

Next, we prove the correctness of Algorithm 1. We start by characterizing the agent's best response.

▶ **Lemma 15.** *On any input (G, δ), when considering the contract problem instance defined in line 3 of Algorithm 1:*

1. *The agent's best response to a contract $0 < \alpha < \frac{M}{M+1+\varepsilon}$ is $\emptyset$.*
2. *The agent's best response to a contract $\frac{M}{M+1+\varepsilon} \leq \alpha < \frac{M}{M+1}$ is a clique of size δ.*
3. *The agent's best response to a contract $\frac{M}{M+1} \leq \alpha < 1$ is a maximum size clique.*

Proof. Let $\alpha \in (0, 1)$.

If $\alpha < \frac{M}{M+1+\varepsilon}$. The agent's utility from a set $S \subseteq V'$ is

$$u_A(\alpha, S) = f(S)\cdot\alpha - |S|\cdot M \leq (M + 1 + \varepsilon)|S|\cdot\alpha - |S|\cdot M,$$

which is strictly negative unless $|S| = 0$.

If $\frac{M}{M+1+\varepsilon} \leq \alpha < \frac{M}{M+1}$. Let T be a clique of size δ in G' (such a clique exists since we can simply take the δ vertices we added to G). The agent's utility from T is

$$u_A(\alpha, T) = f(T)\cdot\alpha - |T|\cdot M = (M+1+\varepsilon)\cdot\delta\cdot\alpha - \delta\cdot M = \delta\left((M+1+\varepsilon)\cdot\alpha - M\right).$$

First, we note that any set S with $|S| > \delta$ is strictly worse than T:

$$\begin{aligned} u_A(\alpha, S) - u_A(\alpha, T) &\leq (|S|(M+1) + \varepsilon\delta)\cdot\alpha - M|S| - \delta\left((M+1+\varepsilon)\cdot\alpha - M\right) \\ &= (|S| - \delta)\left((M+1)\alpha - M\right) < 0. \end{aligned}$$

Additionally, any S such that $|S| \leq \delta$ is not better than T:

$$u_A(\alpha, S) \leq |S|\left((M+1+\varepsilon)\cdot\alpha - M\right) \leq u_A(\alpha, T).$$

These two facts, together with the fact that out of the sets with $|S| \leq \delta$, T maximizes $f(T)$ and the agent breaks ties in favor of the principal, give us that the agent's best response is T (or another clique of size δ).

If $\frac{M}{M+1} \leq \alpha < 1$. Let T be a maximum size clique. The agent's utility from T is

$$u_A(\alpha, T) = f(T) \cdot \alpha - |T| \cdot M = \alpha((M+1)|T| + \varepsilon\delta) - |T| \cdot M \geq \alpha \cdot \varepsilon \cdot \delta.$$

Again, we note that any set S with $|S| > |T|$ is strictly worse than T. Since S cannot be a clique, we get that:

$$u_A(\alpha, S) = \alpha(M|S| + \varepsilon\delta) - |S| \cdot M < \alpha \cdot \varepsilon \cdot \delta$$

Likewise, any set S with $|S| \leq |T|$ isn't better than T:

$$\begin{aligned} u_A(\alpha, S) &\leq (|S|(M+1) + \varepsilon\delta) \cdot \alpha - |S| \cdot M = \varepsilon \cdot \delta \cdot \alpha + |S|((M+1) \cdot \alpha - M) \\ &\leq \varepsilon \cdot \delta \cdot \alpha + |T|((M+1) \cdot \alpha - M) = u_A(\alpha, T). \end{aligned}$$

Again, since T maximizes $f(T)$ (among all sets of size at most $|T|$) and the agent breaks ties in favor of the principal, the the agent's best response is T (or another clique of maximum size). ◀

A direct corollary of Lemma 15 is the following lemma, which is a full characterization of the principal's utility from any contract:

▶ **Corollary 16.** *On any input (G, δ), when considering the contract problem instance defined in line 3 of Algorithm 1:*

1. *The principal's utility from a contract $0 < \alpha < \frac{M}{M+1+\varepsilon}$ is*

$$u_P(\alpha) = 0.$$

2. *The principal's utility from a contract $\frac{M}{M+1+\varepsilon} \leq \alpha < \frac{M}{M+1}$ is*

$$u_P(\alpha) = ((M+1) \cdot \delta + \min(\delta, \delta) \cdot \varepsilon) \cdot (1 - \alpha) = (M + 1 + \varepsilon)\delta(1 - \alpha).$$

3. *The principal's utility from a contract $\frac{M}{M+1} \leq \alpha < 1$ is*

$$u_P(\alpha) = ((M+1)\omega(G') + \delta\varepsilon)\,(1 - \alpha).$$

We are now ready to prove Lemma 13

Proof of Lemma 13. Let (G, δ) be the input of Algorithm 1. To prove Lemma 13 it suffices to prove that if $w(G) \leq \delta$, Algorithm 1 returns SMALL and if $w(G) \geq \frac{2\delta}{\beta^2}$, it returns LARGE.

If $w(G) \leq \delta$. In this case, $\omega(G') = \delta$. Assume towards contradiction that Algorithm 1 returns LARGE, i.e., $\alpha_0 \geq \frac{M}{M+1}$. Since α_0 is a β-approximation, we have

$$u_P(\alpha_0) \geq \beta \cdot u_P\left(\frac{M}{M+1+\varepsilon}\right).$$

Corollary 16 gives us

$$u_P\left(\frac{M}{M+1}\right) \geq u_P(\alpha_0).$$

Putting the two inequalities together, and substituting them with the expressions for the principal's utility from Corollary 16 gives us

$$((M+1)\omega(G') + \delta\varepsilon)\left(1 - \frac{M}{M+1}\right) \geq \beta \cdot (M + 1 + \varepsilon)\delta\left(1 - \frac{M}{M+1+\varepsilon}\right). \tag{2}$$

Note that since $\varepsilon = \frac{2}{\beta} - 1$, then

$$\left(1 - \frac{M}{M+1+\varepsilon}\right) = \frac{1+\varepsilon}{M+1+\varepsilon} = \frac{2}{\beta} \cdot \frac{1}{M+1+\varepsilon} > \frac{2}{\beta} \cdot \frac{1}{2(M+1)} = \frac{1}{\beta} \cdot \left(1 - \frac{M}{M+1}\right),$$

where the inequality is since $M \geq \varepsilon$. Plugging this into Inequality 2 and dividing both sides by $\left(1 - \frac{M}{M+1}\right)$ gives us

$$(M+1)\omega(G') + \delta\varepsilon > (M+1+\varepsilon)\delta.$$

This gives us $\omega(G') > \delta$, contradiction.

If $w(G) \geq \frac{2\delta}{\beta^2}$. In this case, $\omega(G') \geq \frac{2\delta}{\beta^2}$. Assume by contradiction that Algorithm 1 returns SMALL, i.e., $\alpha_0 < \frac{M}{M+1}$. Since α_0 is a β-approximation, we have

$$u_P(\alpha_0) \geq \beta \cdot u_P\left(\frac{M}{M+1}\right).$$

Corollary 16 gives us

$$u_P\left(\frac{M}{M+1+\varepsilon}\right) \geq u_P(\alpha_0).$$

Putting the two inequalities together, and substituting them with the expressions for the principal's utility from Corollary 16 gives us

$$(M+1+\varepsilon)\delta\left(1 - \frac{M}{M+1+\varepsilon}\right) \geq \beta \cdot ((M+1)\omega(G') + \delta\varepsilon)\left(1 - \frac{M}{M+1}\right). \tag{3}$$

Note that since $\varepsilon = \frac{2}{\beta} - 1$, then

$$\left(1 - \frac{M}{M+1+\varepsilon}\right) = \frac{1+\varepsilon}{M+1+\varepsilon} = \frac{2}{\beta} \cdot \frac{1}{M+1+\varepsilon} < \frac{2}{\beta} \cdot \frac{1}{M+1}.$$

Plugging this into the LHS of Inequality 3 and dividing both sides by $\frac{2}{\beta} \cdot \left(1 - \frac{M}{M+1}\right)$ gives us

$$(M+1+\varepsilon)\delta > \frac{\beta^2}{2} \cdot ((M+1)\omega(G') + \delta\varepsilon) \geq \frac{\beta^2}{2} \cdot (M+1+\varepsilon)\omega(G')$$

where the second inequality is from $\omega(G') \geq \delta$. This gives us $\omega(G') < \frac{2\delta}{\beta^2}$, contradiction. ◀

7.2 Using the Differentiation to Approximate $\omega(G)$

In this section we show how to use the guarantees given by Section 7.1 to get a $\beta^2/4$-approximation algorithm for $\omega(G)$, given an oracle access to a β-approximation of the optimal contract, thus concluding the proof of Theorem 11.

▶ **Lemma 17.** *Algorithm 2, given oracle access to a β-approximation of the optimal contract for XOS functions, runs in polynomial time, and on input $G = (V, E)$ gives a $\frac{\beta^2}{4}$-approximation of $\omega(G)$.*

Algorithm 2 Approximation $\omega(G)$.

```
1: Given a graph G = (V, E)
2: for i ← 0 to ⌊log_2(|V|)⌋ do
3:     Run Algorithm 1 on (G, δ = 2^i), and denote its answer by a(i).
4: end for
5: if a(0) = SMALL then
6:     return 1.
7: else
8:     return 2^{i_max} where i_max is the maximal i such that a(i) = LARGE.
9: end if
```

Proof of Lemma 17. If Algorithm 2 returns 1 on line 6, then from Lemma 13, we know that $\omega(G) \leq \frac{2}{\beta^2}$, meaning the algorithm's output is a $\frac{\beta^2}{2}$-approximation, as needed.

Otherwise, let $i_{\max}$ be the maximal i such that $a(i) = \text{LARGE}$. By Lemma 13 we know that $\omega(G) \geq 2^{i_{\max}}$.

If $i_{\max} = \lfloor \log_2(|V|) \rfloor$, returning $2^{i_{\max}} \geq \frac{|V|}{2}$ gives us a $\frac{1}{2}$-approximation (since trivially $\omega(G) \leq |V|$), as needed.

Finally, if $i_{\max} < \lfloor \log_2(|V|) \rfloor$, by our choice of $i_{\max}$ we know that $a(i_{\max} + 1) = \text{SMALL}$, meaning from Lemma 13 that $\omega(G) \leq \frac{2 \cdot 2^{i+1}}{\beta^2}$, which means returning 2^i gives us a $\frac{\beta^2}{4}$-approximation, as needed. ◀

References

1 Tal Alon, Paul Dütting, and Inbal Talgam-Cohen. Contracts with private cost per unit-of-effort. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 52–69, 2021. doi:10.1145/3465456.3467651.

2 Moshe Babaioff, Michal Feldman, and Noam Nisan. Combinatorial agency. In *Proc. ACM EC 2006*, pages 18–28, 2006. doi:10.1016/J.JET.2012.01.010.

3 Yahav Bechavod, Chara Podimata, Steven Wu, and Juba Ziani. Information discrepancy in strategic learning. In *International Conference on Machine Learning*, pages 1691–1715. PMLR, 2022. URL: https://proceedings.mlr.press/v162/bechavod22a.html.

4 Curtis Bechtel, Shaddin Dughmi, and Neel Patel. Delegated pandora's box. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 666–693, 2022. doi:10.1145/3490486.3538267.

5 Matteo Castiglioni, Alberto Marchesi, and Nicola Gatti. Designing menus of contracts efficiently: The power of randomization. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 705–735, 2022. doi:10.1145/3490486.3538270.

6 Matteo Castiglioni, Alberto Marchesi, and Nicola Gatti. Multi-agent contract design: How to commission multiple agents with individual outcomes. In *Proceedings of the 24th ACM Conference on Economics and Computation*, EC '23, pages 412–448. Association for Computing Machinery, 2023. doi:10.1145/3580507.3597793.

7 Krishna Dasaratha, Benjamin Golub, and Anant Shah. Equity pay in networked teams. In *Proceedings of the 24th ACM Conference on Economics and Computation, EC 2023, London, United Kingdom, July 9-12, 2023*, page 512. ACM, 2023. doi:10.1145/3580507.3597754.

8 Paul Dütting, Tomer Ezra, Michal Feldman, and Thomas Kesselheim. Combinatorial contracts. In *Proc. IEEE FOCS 2021*, pages 815–826, 2021. doi:10.1109/FOCS52979.2021.00084.

9 Paul Dütting, Tomer Ezra, Michal Feldman, and Thomas Kesselheim. Multi-agent contracts. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1311–1324, 2023. doi:10.1145/3564246.3585193.

10 Paul Dütting, Michal Feldman, and Yoav Gal Tzur. Combinatorial contracts beyond gross substitutes, 2023. doi:10.48550/ARXIV.2309.10766.

11 Paul Dütting, Tim Roughgarden, and Inbal Talgam-Cohen. The complexity of contracts. *SIAM Journal on Computing*, 50(1):211–254, 2021. doi:10.1137/20M132153X.

12 Yuval Emek and Michal Feldman. Computing optimal contracts in combinatorial agencies. *Theoretical Computer Science*, 452:56–74, 2012. doi:10.1016/J.TCS.2012.05.018.

13 Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998. doi:10.1145/285055.285059.

14 Sanford J. Grossman and Oliver D. Hart. An analysis of the principal-agent problem. *Econometrica*, 51(1):7–45, 1983. doi:10.2307/1912246.

15 Guru Guruganesh, Jon Schneider, and Joshua R Wang. Contracts under moral hazard and adverse selection. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 563–582, 2021. doi:10.1145/3465456.3467637.

16 J. Hastad. Clique is hard to approximate within $n^{1-\varepsilon}$. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 627–636, 1996. doi:10.1109/SFCS.1996.548522.

17 Bengt Holmström. Moral hazard and observability. *The Bell journal of economics*, pages 74–91, 1979. doi:10.2307/3003320.

18 Jon Kleinberg and Robert Kleinberg. Delegated search approximates efficient search. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 287–302, 2018. doi:10.1145/3219166.3219205.

19 Jon Kleinberg and Manish Raghavan. How do classifiers induce agents to invest effort strategically? *ACM Transactions on Economics and Computation (TEAC)*, 8(4):1–23, 2020. doi:10.1145/3417742.

20 Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 18–28, 2001. doi:10.1145/501158.501161.

21 Yingkai Li, Jason D. Hartline, Liren Shan, and Yifan Wu. Optimization of scoring rules. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, pages 988–989. Association for Computing Machinery, 2022. doi:10.1145/3490486.3538338.

22 Noam Nisan and Ilya Segal. The communication requirements of efficient allocations and supporting prices. *Journal of Economic Theory*, 129(1):192–224, 2006. doi:10.1016/J.JET.2004.10.007.

23 Maneesha Papireddygari and Bo Waggoner. Contracts with information acquisition, via scoring rules. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, pages 703–704. Association for Computing Machinery, 2022. doi:10.1145/3490486.3538261.

24 Ramiro Deo-Campo Vuong, Shaddin Dughmi, Neel Patel, and Aditya Prasad. On supermodular contracts and dense subgraphs, 2023. doi:10.48550/ARXIV.2308.07473.

25 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 681–690, 2006. doi:10.1145/1132516.1132612.