Online Contract Design

Elad Lavi[1] , **Hadas Shachnai**[1] and **Inbal Talgam-Cohen**[2]
[1]Computer Science Department, Technion, Haifa, Israel
[2]School of Computer Science, Tel Aviv University
{elad.lavi, hadas}@cs.technion.ac.il, inbaltalgam@gmail.com

Abstract

We initiate the study of online contracts, which integrate the game-theoretic considerations of *economic contract theory*, with the algorithmic and informational challenges of *online algorithm design*. Our starting point is the classic online setting with preemption, in which a hiring principal faces a sequence of adversarial agent arrivals. Upon arrival, the principal must decide whether to tentatively accept the agent to their team, and whether to dismiss previous tentative choices. Dismissal is irrevocable, giving the setting its online decision-making flavor. In our setting, the agents are rational players: once the team is finalized, a game is played where the principal offers contracts (performance-based payment schemes), and each agent decides whether or not to work. Working agents reward the principal, and the goal is to choose a team that maximizes the principal's utility. Our main positive result is a $1/2$-competitive algorithm when agent rewards are additive, which matches the best-possible competitive ratio. Our algorithm is randomized and this is necessary, as we show that no deterministic algorithm can attain a bounded competitive ratio. Moreover, if agent rewards are allowed to exhibit combinatorial structure known as XOS, even randomized algorithms might fail. En route to our competitive algorithm, we develop the technique of *balance points*, which can be useful for further exploration of online contracts in the adversarial model.

1 Introduction

Contract theory studies how a *principal* can incentivize agents to work, when work is costly for the agents and its outcome rewards the principal [Holmström, 1979; Holmström, 1982]. The economic tool for aligning incentives is *contracts* — performance-based payments schemes which the principal commits to. Contract theory is a cornerstone of microeconomics, recognized by the 2016 Nobel prize.

Recently, the rising research area of *algorithmic* contract theory studies contracts through the computational lens (for early papers see e.g. [Babaioff *et al.*, 2012; Dütting *et al.*, 2019], for a survey of the recent flurry of activity see [Dütting *et al.*, 2024]). *Multi-agent* settings are of particular computational interest: In such settings, the principal chooses a subset of agents to work for them as a team, and the agents' contracts must disincentivize them from free-riding on the efforts of other team members. Finding the subset of agents is inherently a combinatorial problem, and is tackled by works like [Babaioff *et al.*, 2012; Dütting *et al.*, 2023a; Deo-Campo Vuong *et al.*, 2024; Castiglioni *et al.*, 2023; Cacciamani *et al.*, 2024; Dütting *et al.*, 2025a; Alon *et al.*, 2025] and many others. To date, the substantial body of work on multi-agent contracts has focused on the *offline* setting, in which the set of agents is known in advance, and the principal optimizes globally.

Bringing algorithmic contract theory closer to real-world applications calls for models that capture how interactions unfold over time. In many classic and modern environments, agents arrive sequentially and unpredictably. For example, a manager heading a team occasionally receives an application from a new potential team member; online labor platforms have a dynamic and fluctuating supply of workers; even when the team is to be composed of AI rather than human agents (see e.g. [Wu *et al.*, 2023]), new AI models appear periodically. How should team contracts be designed when the set of potential participants is revealed gradually over time?

Online algorithms provide a classic framework and rich body of techniques for decision-making in settings characterized by sequential and uncertain arrivals [Borodin and El-Yaniv, 2005]. In this paper, we marry this framework with contracts, and initiate the study of *online contract design*.

1.1 Our Contribution: OMAC Model

As a first step towards integrating the two fields, we formulate the *Online Multi-Agent Contract (OMAC)* model (see Section 2). The OMAC model consists of sequential and uncertain agent arrivals that can conclude abruptly. We briefly survey the design of our model, deferring details to Section 2.

Our first design choice is how to approach uncertainty, where two standard approaches are the adversarial (worst-case) approach, and the Bayesian (stochastic) one. We follow the classic work of [Sleator and Tarjan, 1985], which introduces the *competitive ratio under adversarial inputs* as the gold standard for online algorithms. The competitive ratio measures the relative performance of an online algorithm

compared to an optimal offline one (see Section 2 for a definition). The adversarial approach is particularly well-suited for situations with little to no data.[1] It provides the canonical worst-case baseline for contracts under online arrivals. While our focus is adversarial, we view our work as a gateway to future stochastic (or stochastic-adversarial hybrid) approaches to online contracts (see e.g. [Samuel-Cahn, 1984; Kleinberg, 2005; Babaioff *et al.*, 2007; Kleinberg and Weinberg, 2012]).

We obtain our model by combining two fundamental settings: the adversarial online setting with preemption (e.g., [Buchbinder *et al.*, 2015]), and the multi-agent contract setting [Dütting *et al.*, 2023a]. We now describe these and show how the seemingly unrelated settings blend together naturally.

[Buchbinder *et al.*, 2015] describe an online hiring setting with arbitrarily-arriving agents.[2] The team of agents S must maintain proficiency (as measured by a value function $f(S)$) at all times, since it is unknown when the arrivals will cease, or when the team will need to take action. For example, this is the case if the team's job is to respond to an emergency, recover from a system crash, or supply a surge in demand [Valentine and Bernstein, 2025].

A central feature of their model is preemption — the principal is allowed to replace an agent who was tentatively accepted as more promising agents arrive. Agent dismissal, however, is final and irrevocable — the main characteristic of online decision-making settings. [Buchbinder *et al.*, 2019] show that preemption is key to circumventing intractability and achieving constant-factor guarantees; there are many other such examples (e.g. [Feldman *et al.*, 2009; Han *et al.*, 2015]). Our model displays the same dichotomy, where preemption is what allows meaningful guarantees to emerge; for an impossibility result without preemption, see our Theorem 5.4.

In our model, after an online stage consisting of adversarial agent arrivals, tentative accepts, replacements, and irrevocable dismissals as in [Buchbinder *et al.*, 2019], the composition of the team is finalized. Then, since the agents are rational and utility-maximizing, a game is played: the principal designs and offers contracts to the team members, who in turn decide whether to work or not. A working agent incurs a cost and contributes additively to the principal's reward. This is thus an instantiation with additive rewards of the multi-agent (offline) game formulated by [Dütting *et al.*, 2023a].

As in the formulation of [Dütting *et al.*, 2023a], the principal offers the agents *linear* contracts, i.e., a fixed share of the reward eventually generated by the team. The fixed shares should sum up to at most 1 (the entire reward), or to much less in order to reward the principal — this informally connects the setting to the knapsack problem (for a formal connection to knapsack a.k.a. budgeted maximization, see the full version of this paper [Lavi *et al.*, 2026]).

Luckily, the linear contracts can be designed according to a simple elegant formula of [Babaioff *et al.*, 2012], which induces an equilibrium in which all agents on the team are incentivized to work, while minimizing the principal's overall payment (equivalently, maximizing the principal's total utility). Thus, the main algorithmic challenge in OMAC is to maintain a competitive team during the online stage, while taking into account the overall contractual payment once the team is finalized.

1.2 Our Contribution: Results and Techniques

Working in the OMAC model, in Section 4 we design a randomized algorithm that achieves a competitive ratio of $1/2$ (competitive ratio is the standard approximation measure in online algorithms, see a formal definition in Section 2). We complement our positive result by showing that no deterministic online algorithm achieves a constant competitive ratio. We also establish that randomization allows a competitive ratio of at most $1/2$. The impossibilities are established through a direct adversarial construction (see Section 5). Thus, our randomized algorithm achieves a tight, best-possible guarantee, and the case of additive rewards is fully resolved.

Our tight positive result is obtained via a randomized strategy built around *balance points*, a new structural concept that synchronizes the agents' shares with the principal's utility as arrivals unfold. This technique departs sharply from prior online algorithms, where objectives are always nonnegative and often additive or submodular in the chosen set. In our case, we optimize the principal's incentive-induced utility, which can be negative and non-submodular even when the reward function has additive structure. As a result, existing online algorithms for settings with preemption would fail in our model, as they might output solutions with arbitrarily negative utility. Our approach is explicitly designed to prevent such failures.

To place our approach in context, in the full version of this paper [Lavi *et al.*, 2026] we formalize a connection between our problem and *online budgeted maximization (knapsack) with preemption*, and show how a known randomized algorithm of [Han *et al.*, 2015] for this problem can be adapted to OMAC. However, this algorithm is based on static thresholds, and achieves a $1/4$-competitive ratio for OMAC (as opposed to $1/2$ for knapsack). Balance points serve as dynamic thresholds, which are essential for attaining the optimal competitive ratio of $1/2$. They thus furnish the novel ingredient that resolves the challenge of OMAC optimally.

The connection to online budgeted maximization also reveals a barrier to going beyond additive rewards, to submodular ones: the problem of online monotone submodular maximization subject to a budget constraint remains open [Rawitz and Rosén, 2021]. Beyond submodular there are even more barriers: we can construct instances with XOS rewards (a superclass of submodular), where even randomized strategies fail to secure any constant-factor guarantee — thus ruling out the possibility of meaningful performance in this richer domain (see Section 5). Our construction exploits complementarities among the agents, which cause an agent's value and share to crucially depend on the composition of the team, in a way that is inherently unpredictable in an online setting.

Together, our results show a range of online contract design

[1] This can be the case following technological shifts; e.g., there is little data on the next generation of AI agents to arrive, or on how the worker population will evolve in the GenAI era.

[2] They use somewhat different language than us, referring to candidate players hired by a soccer team manager.

problems, from manageable to intractable. Our resolution of the additive and XOS cases is a first step towards defining the boundary between tractable and hard reward functions in our model of online contracts.

1.3 Related Work

Dynamic Contracts. Temporal models of contracting have previously been explored by multiple works, all quite different from ours. Several works on dynamic contracts have considered settings where a single agent is engaged with repeatedly. E.g., in [Ezra *et al.*, 2024] the agent chooses actions sequentially, and in [Guruganesh *et al.*, 2024] the principal optimizes the contract against a no-regret learning agent. Other works have focused on learning contracts over time, by interacting over many iterations with agents drawn from the same distribution, or alternatively with a single agent. The principal's goal in the learning process is to adapt the incentives based on past outcomes and thus achieve low regret [Ho *et al.*, 2014; Zhu *et al.*, 2023; Bacchiocchi *et al.*, 2024; Chen *et al.*, 2024; Dütting *et al.*, 2023b; Guruganesh *et al.*, 2023]. In [Collina *et al.*, 2024], the principal induces a multi-round game by adaptively choosing among k candidates which agent to contract with, again with the goal of no regret. Reinforcement learning approaches have also been considered [Ivanov *et al.*, 2024; Wu *et al.*, 2024; Bollini *et al.*, 2024].

Preemption and Consistency. Online maximization problems with preemption have been studied extensively and under many different names like *free dismissal*, *free disposal*, *removable*, or *recourse*. A closely related line of work develops *consistent* algorithms, which bound the recourse under dynamic updates in problems like clustering and covering (e.g., [Lattanzi and Vassilvitskii, 2017; Fichtenberger *et al.*, 2021; Gupta and Levin, 2020; Gupta *et al.*, 2022; Łacki *et al.*, 2021; Bhattachary *et al.*, 2024]). In the terminology of [Dütting *et al.*, 2025b], our model is 1-consistent, since each new arrival may cause at most one new hire (dismissed agents are never rehired). However, our algorithmic techniques and analysis are quite different as they are closely tied to the challenges of the contracting model. Closer to our model are works on constrained online submodular maximization. As discussed above, budget constraints are still open; the recent work of [Yang and Zheng, 2024] studies cardinality constraints.

2 The OMAC Model

We introduce an online contract model, which can be viewed from two perspectives: First, as the *online* version of the standard multi-agent contract setting [Babaioff *et al.*, 2012; Dütting *et al.*, 2023a]; second, as the *strategic* version of the classic online decision problem with preemption [Buchbinder *et al.*, 2015; Chan *et al.*, 2018]. We present our model beginning with its online aspects, then its strategic ones.

Online Multi-Agent Contract (OMAC) Setting. In the OMAC setting, a *principal* interacts with an unknown set of *agents* $N = [n]$, revealed sequentially in an arbitrary and unpredictable manner. The principal aims to hire a *team*—a subset $S \subseteq N$ of agents. The team, once finalized, will be tasked with a project. Crucially, the principal does not know the total number of agents n in advance. I.e., from the principal's point of view, the process may terminate after any iteration. The principal must ensure that at every time step, the quality of the current team is competitive.

W.l.o.g., we call the ith agent to arrive agent i, and let $N_i = [i]$ be the first i agents. At every step i, the principal chooses a tentative team S_i, subject to the restriction $S_i \subseteq S_{i-1} \cup \{i\}$ (with the convention that $S_0 = \emptyset$). Note that the team after step i may or may not include the new agent i, and that previously chosen agents in S_{i-1} can be dismissed. However, any agent *not* tentatively chosen (i.e., any agent in $N_{i-1} \setminus S_{i-1}$) cannot be included in S_i. Thus, the principal cannot rehire dismissed agents.

The team S_n chosen in step n is final. I.e., after step n is completed, the solution S is set to S_n, and no additional dismissals are allowed.

The above free dismissal assumption is precisely preemption, studied in many previous works on adversarial online algorithms. Since $|S_i \setminus S_{i-1}| \leq 1$, it is also 1-*consistency*, studied e.g. in [Dütting *et al.*, 2025b] and characteristic of the rapidly growing literature on *stable* online algorithms (see Section 1.3). Without free dismissal, a hired agent would necessarily remain on the team throughout the arrival of subsequent agents. We do not study this variant since, as shown in Section 5.3, it becomes impossible to guarantee any meaningful performance for the principal through an online strategy.

The Principal's Reward Function. The final team S is delegated a project, whose outcome depends on the team's ensemble. As usual in multi-agent contract settings, this is captured by the *reward function* $f : 2^N \to \mathbb{R}_{\geq 0}$, which is a set function specifying the principal's reward $f(S)$ from the project when carried out by team S. As standard in the literature, we assume that f is normalized ($f(\emptyset) = 0$), and monotone ($f(T) \leq f(S)$ whenever $T \subseteq S \subseteq N$). This reflects that a project yields no reward given an empty team, and that extending a team can only increase the principal's reward. Our focus is on *additive* set functions f, where $f(S) = \sum_{i \in S} f(\{i\})$ for all $S \subseteq N$, meaning each agent's contribution is independent of other agents' contributions. In Section 5 we also consider XOS set functions f, for which there exists a family of additive clauses $\{w^{(1)}, \ldots, w^{(m)}\}$ such that $f(S) = \max_{\ell \in [m]} w^{(\ell)}(S)$ for all $S \subseteq N$. In words, f evaluates each team by the most favorable additive clause.

Agent Costs and Linear Contracts (Shares). Each agent i has a known *cost* $c_i \in \mathbb{R}_{\geq 0}$ for working on the project. Thus, for i to work as part of team S, the principal must pay i. In multi-agent contract settings, this is done by *a priori* allocating to i (before the project starts) an α_i-fraction of the eventual reward from the project. This fraction α_i is known as a *linear contract* [Holmström and Milgrom, 1987], and is also referred to as i's *share* [Aharoni *et al.*, 2025]. We write $\alpha(S) = \sum_{i \in S} \alpha_i$ for the total share allocated to team S, with the convention that $\alpha(\emptyset) = 0$.

A defining property of multi-agent contract settings is that once an agent is allocated a fraction of the future reward, this agent might be tempted to *shirk*, allowing other agents to do the costly work while enjoying a fraction of the reward they

generate. This is known in contract design as *moral hazard*.

The agents' shares $(\alpha_i)_{i\in S}$ are designed to overcome moral hazard by making work an equilibrium, while simultaneously ensuring agents are well-compensated and protected in this dynamic setting, mirroring the robust participation incentives of our offline foundation [Babaioff *et al.*, 2012]. When this holds, we say that contract $\boldsymbol{\alpha} = (\alpha_i)_{i\in S}$ *incentivizes* team S. There is a well-known closed formula for shares that incentivize team S, while maximizing the principal's remaining fraction of the project [Babaioff *et al.*, 2012]:

$$\alpha_i := \frac{c_i}{f(S) - f(S \setminus \{i\})}. \tag{1}$$

This α_i represents the minimal share ensuring incentive compatibility: an agent's gain from effort, $\alpha_i(f(S) - f(S \setminus \{i\}))$, must at least cover their cost c_i. When f is additive, α_i simplifies to $c_i / f(\{i\})$.

The Design Problem and Competitive Ratio. We can now describe the full setting: Upon arrival of agent i, the principal learns the cost c_i, and can evaluate the reward function f on any subset of the arrived agents N_i. Based on this information, the principal must immediately decide whether to hire agent i, without knowledge of future arrivals (but with the ability to revoke earlier choices under free dismissal). When agent i is hired, the principal commits to i's share α_i of the eventual reward $f(S)$, where α_i is defined in Eq. (1). This guarantees the agent an incentive to work (despite incurring cost c_i) if they make it to the final team S. The share α_i is parameterized by S until step n, when S is finalized. Once an agent is dismissed, their share is set to zero.

The players' utilities are as follows: The utility of each agent $i \in S$ is $\alpha_i f(S) - c_i$, the utility of any other agent is 0, and the principal's utility depends on the remaining share, and is given by $g(S) := \big(1 - \alpha(S)\big) f(S)$.

Since the agents' shares $(\alpha_i)_{i\in S}$ are determined by S, the principal's design problem boils down to an online algorithm for selecting S such that the principal's utility $g(S)$ is maximized. In online optimization, the performance of an online algorithm is typically evaluated using the *competitive ratio* (CR), which compares the outcome of the online algorithm to that of an optimal offline algorithm that has complete knowledge of the input sequence in advance.

Formally, an online algorithm ALG has a CR $\gamma \in [0,1]$ if for every input sequence I, its output satisfies $\mathsf{ALG}(I) \geq \gamma \cdot \mathsf{OPT}(I)$, where $\mathsf{OPT}(I)$ denotes the value achieved by the optimal offline solution. When the algorithm is randomized and the adversary is oblivious (i.e., cannot adapt to the random choices), the *expected competitive ratio* is a random variable defined similarly, requiring that $\mathbf{E}[\mathsf{ALG}(I)] \geq \gamma \cdot \mathsf{OPT}(I)$ for all I. In our model, $\mathsf{OPT}(N_i) = \max_{S_i \subseteq N_i} g(S_i)$ for every $i \in [n]$.

3 Toolbox: Balance Points

In this section, we introduce the notion of *balance points*—a key concept that captures the trade-off between allocating additional shares to agents to incentivize work, and preserving the principal's utility—forming the basis for both the design and analysis of our algorithm in Section 4.

3.1 Agent Quality

We start by defining a quality measure for agents and teams, which captures economic efficiency, namely, the value generated by the hired agents, relative to the total share that must be allocated to them in order to incentivize their work.

Definition 3.1 (Quality)**.** *Consider an additive reward function f. For every team $S \subseteq N$, its* quality $q(S)$ *is the ratio between the reward it yields, and the total share allocated to the team members:*

$$q(S) := \frac{f(S)}{\alpha(S)} = \frac{f(S)}{\sum_{i\in S} \frac{c_i}{f(\{i\})}}; \quad q(\emptyset) := 0. \tag{2}$$

Definition 3.1 applied to a singleton team $\{i\}$ gives the quality of agent i: $q_i := \frac{f(\{i\})}{\alpha_i} = \frac{f^2(\{i\})}{c_i}$. Using Eq. (2) we can rewrite the principal's utility using the notion of quality:

$$g(S) = (1 - \alpha(S))\alpha(S)q(S). \tag{3}$$

Notation. We partition the set of all agents N into subsets of uniform quality, ordered from highest to lowest: $N = Q_1 \cup \cdots \cup Q_p$, where for every $1 \leq x \leq p$, all agents in Q_x have the same quality, which is strictly greater than that of any agent in Q_y with $x < y \leq p$. Each Q_x is referred to as a *quality group*, and we denote its uniform quality by q^x; thus, $q^x = q_i$ for every agent $i \in Q_x$ and $q^x = q(Q_x)$.

Since f is additive, each agent i has a fixed value α_i that can be determined upon arrival. Consequently, an agent's quality and associated quality group remain unchanged throughout the process.

Consider a hired set of agents $S \subseteq N$. For every $1 \leq x \leq p$, let $S^x := S \cap \bigcup_{1\leq y\leq x} Q_y$ be the subset of S consisting of agents from the first x quality groups, with $S^0 := \emptyset$.

Let $\bar{S}$ denote the ordering of agents in S in decreasing order of quality. In case of ties in quality, agents with smaller shares appear earlier in the ordering. Let $\bar{N}$ denote the entire set of agents, ordered in this way. For every $i \in [n]$, let $\bar{N}_i = (\bar{N})_i$ denote the agents in $\bar{N}$ appearing up to agent i, including i itself. Now we can define $\bar{S}_i := S \cap \bar{N}_i$. In words, these are the agents in $\bar{S}$ that are ordered before agent i.[3]

3.2 Balance Point Definition

Intuitively, a balance point is defined per quality group Q_x and per team S (which determines the set S^{x-1} of higher-quality agents who are allocated shares). It quantifies the maximum additional share that can be allocated to agents of quality Q_x before the principal's utility begins to decline. Once the balance point is exceeded by the allocated shares, adding more agents from Q_x or from lower-quality groups can only decrease the principal's utility. Therefore, identifying and maintaining allocations near the balance point is key to achieving optimal utility when selecting agents in an online manner.

Definition 3.2 (Balance point)**.** *Given a hired set of agents $S \subseteq N$ and a quality group Q_x, the* balance point $b(S,x) \in$

[3]For example, if the agents arrive in increasing order of quality, and $S = N$, then $\bar{S}_n = \{n\}$ since all other agents succeed agent n in the ordering.

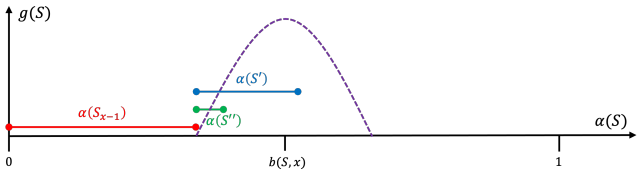

Figure 1: This figure illustrates the geometric intuition for balance points. For a team S and quality level x, the balance point $b(S, x)$ represents allocated share. Recall that S^{x-1} is the subset of agents of quality at least q^{x-1}. For two subsets $S', S'' \subseteq Q_x$ of quality q^x, the horizontal segments show the shares allocated to $S^{x-1} \cup S'$ (red and blue) and $S^{x-1} \cup S''$ (red and green). Since the share of $S^{x-1} \cup S'$ is closer to the balance point than that of $S^{x-1} \cup S''$, by Definition 3.2, $g(S^{x-1} \cup S') \geq g(S^{x-1} \cup S'')$. The y-axis represents the principal's utility, and the dashed parabola depicts the utility as a function of shares allocated to agents of quality q^x. The principal's utility is maximized at the balance point, and indeed $g(S^{x-1} \cup S') \geq g(S^{x-1} \cup S'')$.

$(0, 1)$ *of* Q_x *relative to* S *is a share satisfying the following property. For any two subsets* $S', S'' \subseteq Q_x$ *such that* $\left|\alpha(S^{x-1} \cup S') - b(S, x)\right| \leq \left|\alpha(S^{x-1} \cup S'') - b(S, x)\right|$*, it holds that*

$$g(S^{x-1} \cup S') \geq g(S^{x-1} \cup S'').$$

Figure 1 illustrates this definition geometrically. In Lemma 3.4 we show that such a balance point exists for every quality group and every set of hired agents.

To better understand the idea of balance points, consider a fixed quality group, Q_x. Similarly to Eq. (3), we can use the additivity of f to write the principal's utility as

$$g(S) = \big(1 - \alpha(S)\big)\left(\alpha(S \cap Q_x) \cdot q^x + \alpha(S \setminus Q_x) \cdot q(S \setminus Q_x)\right).$$

Suppose that the agents hired from other quality groups $S \setminus Q_x$ are fixed. As the principal allocates more shares to agents in Q_x, the utility may increase at first but is guaranteed to decline eventually. Indeed, by the monotonicity of f, hiring additional agents from Q_x (increasing $\alpha(S \cap Q_x)$) increases the reward $f(S \cap Q_x) = \alpha(S \cap Q_x)q^x$ contributed by this quality group, thereby increasing the total reward. At the same time, the principal's share of the reward $1 - \alpha(S \setminus Q_x) - \alpha(S \cap Q_x)$ is reduced. Thus, the marginal effect of hiring agents in Q_x on $g(S)$ follows a concave trajectory as a function of $\alpha(S \cap Q_x)$, forming a downward-opening parabola with the balance point marking its peak—see Figure 1.

3.3 Characterizing Balance Points

We now give an alternative characterization of balance points using an *auxiliary agent*: a hypothetical agent of quality q^x, whose share is set according to Eq. (1) by setting a corresponding cost. The auxiliary agent whose share maximizes utility relative to S^{x-1} yields the balance point $b(S, x)$.

Observation 3.3. *Let* r *be an auxiliary agent with* $q_r = q^x$ *such that* $g(S^{x-1} \cup \{r\}) \geq g(S^{x-1} \cup \{v\})$ *for any other auxiliary agent* v *with* $q_v = q^x$*. Then,* $\alpha(S^{x-1} \cup \{r\}) = b(S, x)$*.*

The concept of auxiliary agent gives a simple and intuitive way to derive the balance point of each quality group, as it represents the optimal share allocation to agents of a given quality. This allows us to express $b(S, x)$ using a concise algebraic formula.

Lemma 3.4. *For every hired set of agents* $S \subseteq N$ *and quality group* Q_x*,*

$$b(S, x) = \frac{1}{2} + \frac{1}{2}\left(1 - \frac{q(S^{x-1})}{q^x}\right)\alpha(S^{x-1}). \tag{4}$$

Corollary 3.5. *For every* $S \subseteq N$*, the balance point of the first quality group is* $b(S, 1) = 1/2$*.*

3.4 Comparing Hired Agent Sets

We use balance points to compare sets of hired agents based on the total utility they generate for the principal. Specifically, if a set allocates shares more effectively across quality groups while staying within its balance points, its total utility is guaranteed to be higher.

Lemma 3.6. *Let* $S, S' \subseteq N$ *be two sets of hired agents. Suppose that for every quality group* Q_x*,* $1 \leq x \leq p$*, the shares satisfy* $\alpha(S \cap Q_x) \leq \alpha(S' \cap Q_x)$ *and* $\alpha(S^x) \leq \alpha((S')^x) \leq b(S', x)$*, then it holds that* $g(S) \leq g(S')$*.*

The statement of the above lemma naturally extends to the case where we introduce an auxiliary agent r that maximizes the contribution of a given quality group, assuming the agents in all preceding quality groups are fixed.

Lemma 3.7. *Let* $S, S' \subseteq N$ *be two sets of hired agents and* $1 \leq y \leq p$*. Suppose that for every quality group* Q_x*,* $1 \leq x \leq y$*, the shares satisfy* $\alpha(S \cap Q_y) \leq \alpha(S' \cap Q_y)$ *and* $\alpha(S^y) \leq \alpha((S')^y) \leq b(S', y)$*. Then* $g(S) \leq g((S')^y \cup \{r\})$*, where* r *is the auxiliary agent that maximizes utility over the fixed* $(S')^y$*.*

4 Positive Result: Randomized Algorithm

In this section, we present our randomized algorithm for OMAC. We start with two deterministic algorithms representing complementary approaches to OMAC with an additive reward function. Our final algorithm is a randomized combination of these strategies.

Algorithm BP (Balance Point) hires agents with the highest possible utility while ensuring no group exceeds its balance points. When comparing two agents of equal quality, BP prefers the one with the smaller share. To achieve this, it runs BP-STEP (Algorithm 1) upon each agent's arrival.

Algorithm MAX selects the best agent from all arrivals by running MAX-STEP upon each agent's arrival. It is formally described in Algorithm 2. For simplicity, we initialize with a dummy agent 0, having $\alpha_0 = 0$, indicating no agents have been hired yet.

We define the randomized algorithm ALG-OMAC (Algorithm 3) as follows: With equal probability (i.e., uniformly at random), run either BP or MAX throughout the entire arrival process. The following illustrates the executions of BP and MAX on a short arrival sequence.

Example 4.1. *Let* $N = \{A, B, C, D\}$ *with* $\alpha_A = 0.05$ *and* $f(\{A\}) = 0.15$*,* $\alpha_B = 0.25$ *and* $f(\{B\}) = 1.00$*,* $\alpha_C = 0.16$ *and* $f(\{C\}) = 0.80$*,* $\alpha_D = 0.35$ *and* $f(\{D\}) = 1.75$*. By*

Algorithm 1 BP-STEP

Input: A previously hired group of agents S, and an arriving agent j.

$S' \leftarrow S \cup \{j\}$
Sort the agents in S' in decreasing order of quality, breaking ties by non-decreasing order of shares.
Let $\bar{S}' = \{i_1, \ldots, i_s\}$ be the sorted list.
Set $S \leftarrow \emptyset$
for $k = 1$ to s **do**
  Let x be such that $i_k \in Q_x$
  if $\alpha(S \cup \{i_k\}) < b(S, x)$ **then**
    $S \leftarrow S \cup \{i_k\}$
  end if
end for
return S

Algorithm 2 MAX-STEP

Input: A previously hired agent i, and an arriving agent j.

return $\arg\max\{g(\{i\}), g(\{j\})\}$

evaluating agent qualities, $q_i = f(\{i\})/\alpha_i$, we have three quality groups: Q_1 ($q^1 = 5$) contains C and D, Q_2 ($q^2 = 4$) contains B, and Q_3 ($q^3 = 3$) contains A.

We briefly trace the execution of Algorithm BP *upon each arrival.*

Arrival of A: *The tentative team is $S = \{A\}$. As $\alpha_A \leq 0.5 = b(\emptyset, 3)$,* BP *tentatively hires A.*

Arrival of B: *The sorted quality order is B, A.* BP *keeps B as $\alpha_B \leq 0.5$. For A, the dynamic threshold drops to $b(\{B\}, 3) \approx 0.458$; since $\alpha(\{B, A\}) = 0.30 \leq 0.458$, A is kept, and $S = \{B, A\}$.*

Arrival of C: *The order is C, B, A. C is kept as $\alpha_C \leq 0.5$. For B, $b(\{C\}, 2) = 0.5 + 0.5(1 - \frac{5}{4})0.16 = 0.48$, and since $\alpha(\{C, B\}) = 0.41 \leq 0.48$, B is kept. For A, the threshold is $b(\{C, B\}, 3) = 0.405$. The total share satisfies $\alpha(\{C, B\}) = 0.41 > 0.405$; therefore, A is irrevocably dismissed, and $S = \{C, B\}$.*

Arrival of D: *The order within Q_1 is $\{C, D\}$. C is evaluated first and kept. For D, the threshold is $b(\{C\}, 1) = 0.5$. Since $\alpha(\{C, D\}) = 0.51 > 0.5$, D is rejected. B evaluates safely against C as before.*

The final team selected by BP *is $\{C, B\}$, obtaining the utility $g(\{C, B\}) = 1.062$. Simultaneously, Algorithm* MAX *selects $\{D\}$, the team yielding the highest utility from a single agent. Hence, the utility of* MAX *is $g(\{D\}) = 1.1375$.*

The optimal offline team for this instance is $OPT = \{D, A\}$, yielding $g(\{D, A\}) = 1.14$. Notably, this optimal subset violates the balance point conditions. By adhering strictly to these thresholds, BP *guarantees a secure utility against adversarial worst-cases, while* MAX *captures heavy-hitting individual agents.*

Algorithm 3 ALG-OMAC

Input: An instance of OMAC.

With probability $1/2$, run BP; with probability $1/2$, run MAX.

We now formally analyze ALG-OMAC.

Theorem 4.2. *Given an instance I of OMAC, it holds that* $\mathsf{OPT}(I) \leq g(\mathsf{BP}(I)) + g(\mathsf{MAX}(I))$.

Corollary 4.3. *The expected CR of* ALG-OMAC *is* $\geq 1/2$.

Proof. By the definition of ALG-OMAC, its expected utility is given by $\frac{1}{2}\left(g(\mathsf{BP}(I)) + g(\mathsf{MAX}(I))\right)$. Therefore, by Theorem 4.2, the competitive ratio of the algorithm is $\geq 1/2$. □

We proceed to formally analyze the algorithm. Let S^1 and S^2 denote the groups of agents hired by BP and MAX, respectively, given the arrival sequence N. Let $\bar{S}^* = \{1^*, \ldots, k^*\}$ denote an optimal solution of size $k = |S^*|$, where agents are sorted in decreasing order of quality, with agents of the same quality ordered in non-decreasing order of shares.

Let $t \in Q_z$ be the first agent that satisfies at least one of these two properties:

- $\alpha(\bar{N}_t) \geq b(N, z)$ and $\alpha_t \leq 1/2$;
- $t = k^*$.

In other words, t is defined as the first agent in the ordering N_t who, when hired together with all preceding agents, crosses their corresponding balance point and whose share size is at most $1/2$—or, alternatively, as the last agent in $\bar{S}^*$, whichever occurs earlier. Denote by t' the agent that immediately precedes t in the ordering.

Lemma 4.4. $\bar{N}_{t'} \subseteq S^1$.

We note that containment (between two subsets $S, S' \subseteq N$) is a special case of Lemma 3.6. Hence, using Lemma 4.4 and the fact that $\bar{S}^*_{t'} \subseteq \bar{N}_{t'}$, we have

$$g(\bar{S}^*_{t'}) \leq g(\bar{N}_{t'}) \leq g(\bar{S}^1). \tag{5}$$

Proof of Theorem 4.2: We distinguish between two cases.

Assume first that t is the last agent in S^*. By definition, $S^* = \bar{S}^*_{t'} \cup \{t\}$, and by the sub-additivity of g, we have

$$g(S^*) = g(\bar{S}^*_{t'} \cup \{t\}) \leq g(\bar{S}^*_{t'}) + g(\{t\}).$$

By inequality (5), $g(\bar{S}^*_{t'}) \leq g(S^1)$, and from the maximality of the single agent selected by algorithm MAX, we have $g(\{t\}) \leq g(S^2)$. Therefore,

$$g(S^*) \leq g(\bar{S}^*_{t'}) + g(\{t\}) \leq g(S^1) + g(S^2).$$

If t is not last in S^* then it must hold that $\alpha_t \leq 1/2$. We know that $S^{*z-1} \subseteq N^{z-1}$, which allows us to apply Lemma 3.7. Consequently, $g(S^*) \leq g(N^{z-1} \cup \{r\})$, where r denotes the auxiliary agent corresponding to N^z. By sub-additivity of g, we have that $g(S^*) \leq g(N^{z-1}) + g(\{r\})$. Furthermore, using Lemma 3.6, we have $N^{z-1} \subseteq \bar{N}_{t'}$, implying that $g(N^{z-1}) \leq g(\bar{N}_{t'})$. Thus, $g(S^*) \leq g(\bar{N}_{t'}) + g(\{r\})$.

Now, observe that in this case, agent t crosses its balance point over $\bar{N}_{t'}$, i.e., $\alpha(\bar{N}_t) \geq b(N, z)$. As a result, we have by Corollary 3.5

$$\alpha(\bar{N}_{t'}) + \alpha_t = \alpha(\bar{N}_{t'} \cup \{t\}) \geq b(N, z) = \alpha(\bar{N}_{t'}) + \alpha_r,$$

and hence $\alpha_t \geq \alpha_r$.

Since $\alpha_r \leq \alpha_t \leq 1/2$, $g(\{r\}) \leq g(\{t\})$. Moreover, as $t \in N$ is not an auxiliary agent (unlike r), MAX ensures that $g(\{t\}) \leq g(S^2)$. Therefore, $g(\{r\}) \leq g(S^2)$.

Finally, from inequality (5), we also have $g(\bar{N}_{t'}) \leq g(S^1)$. Combining these bounds, we conclude that

$$g(S^*) \leq g(\bar{N}_{t'}) + g(\{r\}) \leq g(S^1) + g(S^2). \quad \square$$

5 Negative Results

5.1 Additive Rewards

We identify two fundamental limitations in the additive case, both arising from the same adversarial scenario: the arrival of an agent demanding a large share. Such an agent can either cause the principal to lose all project value, or block the recruitment of more effective agents. Consequently, deterministic strategies might perform arbitrarily poorly.

Theorem 5.1. *The CR of any deterministic algorithm for OMAC can be arbitrarily small.*

Furthermore, even randomized algorithms cannot fully avoid the trade-off induced by such an agent: no randomized algorithm can achieve a competitive ratio better than $1/2$ (our randomized algorithm in Section 4 matches this bound).

Theorem 5.2. *No (randomized) algorithm for OMAC can achieve a CR* $> 1/2$.

Proof sketch. Let $\varepsilon \in (0, 1)$ be a small value, $n = \varepsilon^2/2 + 1$ and $N = \{1, \ldots, n\}$ such that $\alpha_1 = 1 - \varepsilon^2, q_1 = \varepsilon$ and $\alpha_i = \varepsilon^2, q_i = \varepsilon^2$ for every $2 \leq i \leq n$.

Given a (possibly randomized) algorithm, let X be an indicator random variable for hiring agent 1 upon arrival. For any small agent $i \in N \setminus 1$, let $T_i \subseteq N_i \setminus 1$ be the random set of small agents hired by the algorithm among the first i arrivals, conditioned on hiring agent 1, with $t_i = |T_i|$.

Then every randomized algorithm belongs to one of the following types:

1. ALG_1 is the class of algorithms for which the corresponding indicator variable X satisfies $\Pr(X) \leq 1/2$.
2. For all $i \in N \setminus \{1\}$, let ALG_i be the class of algorithms for which $\Pr(X) > 1/2$. In addition, i is the first index such that $\mathbf{E}[t_i] > 1$, meaning that for every $1 \leq j < i$, we have $\mathbf{E}[t_j] \leq 1$.
3. ALG_{n+1} denotes the class of algorithms for which $\Pr(X) > 1/2$, and $\mathbf{E}[t_i] \leq 1$ for all $1 \leq i \leq n$.

Consider one of the classes ALG_i for $1 \leq i \leq n+1$. We claim that by setting $k = \min\{i, n\}$, the utility of every algorithm in this class on the input prefix N_k can be made arbitrarily small compared to the optimal solution. In [Lavi *et al.*, 2026], we prove this claim separately for each class. $\square$

5.2 XOS Rewards

We study OMAC with an XOS reward function f and show that, even under free dismissal, no (possibly randomized) algorithm can achieve a constant competitive ratio. This impossibility stems from the adversary's ability to nullify an agent's marginal contribution as new agents arrive. As in real economic settings, the principal lacks foresight into future interactions, making reliable long-term decisions impossible.

Theorem 5.3. *For OMAC with an XOS reward function f, the CR of any algorithm can be arbitrarily small.*

Proof sketch. We apply Yao's min–max principle using a distribution $\mathcal{D}$ over m sequential groups of n indistinguishable agents. Each group contains exactly one "good" agent and $n - 1$ "bad" agents, where the class of each agent cannot be revealed for the principal upon arrival. Within any group, agent i provides zero marginal contribution in their team S, leading to $\alpha_i = \infty$ and $g(S) = -\infty$ if multiple agents are hired. Thus, productive collaboration occurs exclusively between good agents from different groups.

Ex ante, ALG hires the good agent with probability only $1/n$. Failure to identify a good agent precludes synergy with future arrivals, forcing a dilemma between retaining a likely bad hire or swapping for another uncertain candidate. While the optimal offline solution pairs all m good agents, the algorithm's expected utility remains suppressed by the low success rate per group. As both n and m increase, the expected competitive ratio goes to zero.

In [Lavi *et al.*, 2026], we give an example and clarify the construction of the distribution $\mathcal{D}$ along with the behavior of algorithms in this setting. $\square$

5.3 Without Preemption

To complete the picture, we show that dismiss-free OMAC admits no algorithm with a competitive ratio (CR) bounded away from zero. This impossibility holds even under the assumption of uniform agent quality (see Definition 3.1). The impossibility follows by taking $n \to \infty$ in the next theorem:

Theorem 5.4. *Consider OMAC with uniform agent quality q and at most n arriving agents. The CR γ of any algorithm* ALG *satisfies $\gamma = O(1/n)$.*

We give a formal proof in [Lavi *et al.*, 2026].

6 Future Work

We initiate the study of online team formation with rational team members who must be incentivized by contracts. We completely resolve the case of additive rewards by developing the technique of balance points. We highlight a gap in the literature — online algorithms for budgeted submodular maximization. A first step towards bridging this gap can be online maximization of weighted matroid rank functions subject to a budget constraint. Our work focuses on establishing the adversarial baseline; a natural direction for future work is to explore stochastic refinements — from secretary models to prophet inequalities — to achieve stronger performance guarantees under random arrival patterns.

Acknowledgments

We gratefully acknowledge feedback from anonymous reviewers that led to this improved version of our work. This work was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 101077862, project ALGOCONTRACT), by the Israel Science Foundation (grant No. 3331/24), by the NSF-BSF (grant No. 2021680), and by a Google Research Scholar Award.

References

[Aharoni *et al.*, 2025] Gil Aharoni, Martin Hoefer, and Inbal Talgam-Cohen. Welfare and beyond in multi-agent contracts. In *Proc. EC 2025*, page 895. ACM, 2025.

[Alon *et al.*, 2025] Tal Alon, Matteo Castiglioni, Junjie Chen, Tomer Ezra, Yingkai Li, and Inbal Talgam-Cohen. Multi-project contracts. In *EC*, page 580–598, 2025.

[Babaioff *et al.*, 2007] Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, secretary problems, and online mechanisms. In *SODA*, pages 434–443. SIAM, 2007.

[Babaioff *et al.*, 2012] Moshe Babaioff, Michal Feldman, Noam Nisan, and Eyal Winter. Combinatorial agency. *Journal of Economic Theory*, 147(3):999–1034, 2012.

[Bacchiocchi *et al.*, 2024] Francesco Bacchiocchi, Matteo Castiglioni, Alberto Marchesi, and Nicola Gatti. Learning optimal contracts: How to exploit small action spaces. In *ICLR*, 2024.

[Bhattachary *et al.*, 2024] Sayan Bhattachary, Martin Costa, Naveen Garg, Silvio Lattanzi, and Nikos Parotsidis. Fully dynamic k-clustering with fast update time and small recourse. In *FOCS*, pages 216–227, 2024.

[Bollini *et al.*, 2024] Matteo Bollini, Francesco Bacchiocchi, Matteo Castiglioni, Alberto Marchesi, and Nicola Gatti. Contracting with a reinforcement learning agent by playing trick or treat. Available at https://arxiv.org/abs/2410.13520, 2024.

[Borodin and El-Yaniv, 2005] Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 2005.

[Buchbinder *et al.*, 2015] Niv Buchbinder, Moran Feldman, and Roy Schwartz. Online submodular maximization with preemption. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1202–1216, 2015.

[Buchbinder *et al.*, 2019] Niv Buchbinder, Moran Feldman, and Roy Schwartz. Online submodular maximization with preemption. *ACM Trans. Algorithms*, 15(3):30:1–30:31, 2019.

[Cacciamani *et al.*, 2024] Federico Cacciamani, Martino Bernasconi, Matteo Castiglioni, and Nicola Gatti. Multi-agent contract design beyond binary actions. In *EC*, page 1293, 2024.

[Castiglioni *et al.*, 2023] Matteo Castiglioni, Alberto Marchesi, and Nicola Gatti. Multi-agent contract design: How to commission multiple agents with individual outcome. In *EC*, pages 412–448, 2023.

[Chan *et al.*, 2018] T.-H. Hubert Chan, Zhiyi Huang, Shaofeng H.-C. Jiang, Ning Kang, and Zhihao Gavin Tang. Online submodular maximization with free disposal. *ACM Trans. Algorithms*, 14(4):56:1–56:29, 2018.

[Chen *et al.*, 2024] Yurong Chen, Zhaohua Chen, Xiaotie Deng, and Zhiyi Huang. Are bounded contracts learnable and approximately optimal? In *EC*, 2024.

[Collina *et al.*, 2024] Natalie Collina, Varun Gupta, and Aaron Roth. Repeated contracting with multiple nonmyopic agents: Policy regret and limited liability. In *EC*, pages 640–668, 2024.

[Deo-Campo Vuong *et al.*, 2024] Ramiro Deo-Campo Vuong, Shaddin Dughmi, Neel Patel, and Aditya Prasad. On supermodular contracts and dense subgraphs. In *SODA*, pages 109–132, 2024.

[Dütting *et al.*, 2019] Paul Dütting, Tim Roughgarden, and Inbal Talgam-Cohen. Simple versus optimal contracts. In *EC*, pages 369–387, 2019.

[Dütting *et al.*, 2023a] Paul Dütting, Tomer Ezra, Michal Feldman, and Thomas Kesselheim. Multi-agent contracts. In *STOC*, pages 1311–1324, 2023.

[Dütting *et al.*, 2023b] Paul Dütting, Guru Guruganesh, Jon Schneider, and Joshua Wang. Optimal no-regret learning of one-sided Lipschitz functions. In *ICML*, 2023.

[Dütting *et al.*, 2024] Paul Dütting, Michal Feldman, and Inbal Talgam-Cohen. Algorithmic contract theory: A survey. *Foundations and Trends in Theoretical Computer Science*, 16(3-4):211–412, 2024.

[Dütting *et al.*, 2025a] Paul Dütting, Tomer Ezra, Michal Feldman, and Thomas Kesselheim. Multi-agent combinatorial contracts. In *SODA*, pages 1857–1891, 2025.

[Dütting *et al.*, 2025b] Paul Dütting, Federico Fusco, Silvio Lattanzi, Ashkan Norouzi-Fard, Ola Svensson, and Morteza Zadimoghaddam. The cost of consistency: Submodular maximization with constant recourse. In *STOC*, pages 1406–1417. ACM, 2025.

[Ezra *et al.*, 2024] Tomer Ezra, Michal Feldman, and Maya Schlesinger. Contract design for sequential actions. *arXiv:2403.09545*, 2024.

[Feldman *et al.*, 2009] Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S. Muthukrishnan. Online ad allocation with free disposal. In *ESA*, pages 374–385. Springer, 2009.

[Fichtenberger *et al.*, 2021] Hendrik Fichtenberger, Silvio Lattanzi, Ashkan Norouzi-Fard, and Ola Svensson. Consistent k-clustering for general metrics. In *SODA*, pages 2660–2678. SIAM, 2021.

[Gupta and Levin, 2020] Anupam Gupta and Roie Levin. Fully-dynamic submodular cover with bounded recourse. In *FOCS*, pages 1147–1157. IEEE, 2020.

[Gupta *et al.*, 2022] Anupam Gupta, Vijaykrishna Gurunathan, Ravishankar Krishnaswamy, Amit Kumar, and Sahil Singla. Online discrepancy with recourse for vectors and graphs. In *SODA*, pages 1356–1383. SIAM, 2022.

[Guruganesh *et al.*, 2023] Guru Guruganesh, Jon Schneider, Joshua Wang, and Junyao Zhao. The power of menus in contract design. In *EC*, pages 818–848, 2023.

[Guruganesh *et al.*, 2024] Guru Guruganesh, Yoav Kolumbus, Jon Schneider, Inbal Talgam-Cohen, Emmanouil-Vasileios Vlatakis-Gkaragkounis, Joshua Wang, and S. Matthew Weinberg. Contracting with a learning agent. *Advances in Neural Information Processing Systems*, 37:77366–77408, 2024.

[Han *et al.*, 2015] Xin Han, Yasushi Kawase, and Kazuhisa Makino. Randomized algorithms for removable online knapsack problems. *Theoretical Computer Science*, 562:395–405, 2015.

[Ho *et al.*, 2014] Chien-Ju Ho, Aleksandrs Slivkins, and Jennifer Wortman Vaughan. Adaptive contract design for crowdsourcing markets: Bandit algorithms for repeated principal-agent problems. In *EC*, pages 359–376, 2014.

[Holmström and Milgrom, 1987] Bengt Holmström and Paul Milgrom. Aggregation and linearity in the provision of intertemporal incentives. *Econometrica*, pages 303–328, 1987.

[Holmström, 1979] Bengt Holmström. Moral hazard and observability. *The Bell Journal of Economics*, pages 74–91, 1979.

[Holmström, 1982] Bengt Holmström. Moral hazard in teams. *The Bell Journal of Economics*, 13(2):324–340, 1982.

[Ivanov *et al.*, 2024] Dima Ivanov, Paul Dütting, Inbal Talgam-Cohen, Tonghan Wang, and David C Parkes. Principal-agent reinforcement learning: Orchestrating AI agents with contracts. *arXiv:2407.18074*, 2024.

[Kleinberg and Weinberg, 2012] Robert Kleinberg and S. Matthew Weinberg. Matroid prophet inequalities. In *STOC*, pages 123–136. ACM, 2012.

[Kleinberg, 2005] Robert Kleinberg. A multiple-choice secretary problem with applications to online auctions. In *SODA*, pages 630–631. SIAM, 2005.

[Łacki *et al.*, 2021] Jakub Łacki, Bernhard Haeupler, Christoph Grunau, Rajesh Jayaram, and Václav Rozhoň. Fully dynamic consistent k-center clustering. In *SODA*, pages 3463–3484. SIAM, 2021.

[Lattanzi and Vassilvitskii, 2017] Silvio Lattanzi and Sergei Vassilvitskii. Consistent k-clustering. In *ICML*, pages 1975–1984, 2017.

[Lavi *et al.*, 2026] Elad Lavi, Hadas Shachnai, and Inbal Talgam-Cohen. Online contract design. *arXiv preprint arXiv:2602.07385*, 2026.

[Rawitz and Rosén, 2021] Dror Rawitz and Adi Rosén. Online budgeted maximum coverage. *Algorithmica*, 83(9):2989–3014, 2021.

[Samuel-Cahn, 1984] Esther Samuel-Cahn. Comparison of threshold stop rules and maximum for independent nonnegative random variables. *The Annals of Probability*, 12(4):1213–1216, 1984.

[Sleator and Tarjan, 1985] Daniel Dominic Sleator and Robert Endre Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985.

[Valentine and Bernstein, 2025] Melissa Valentine and Michael S Bernstein. *Flash Teams: Leading the Future of AI-Enhanced, On-Demand Work*. MIT Press, 2025.

[Wu *et al.*, 2023] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversation framework. https://arxiv.org/abs/2308.08155, 2023.

[Wu *et al.*, 2024] Jibang Wu, Siyu Chen, Mengdi Wang, Huazheng Wang, and Haifeng Xu. Contractual reinforcement learning: Pulling arms with invisible hands, 2024. arXiv preprint.

[Yang and Zheng, 2024] Zhengchen Yang and Jiping Zheng. Online submodular maximization via adaptive thresholds. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI*, pages 7065–7073, 2024.

[Zhu *et al.*, 2023] Banghua Zhu, Stephen Bates, Zhuoran Yang, Yixin Wang, Jiantao Jiao, and Michael I. Jordan. The sample complexity of online contract design. In *EC*, page 1188, 2023.