Algorithmic Contract Design for Teams

Michal Feldman

Yoav Gal-Tzur

Tomek Ponitka

Maya Schlesinger

Tel Aviv University

EC 2026, Rome, Italy

Tutorial Outline

Part I

Motivation and background
Multi-agent binary-action model
Key challenges and techniques

Part II

Multi-agent multi-action model
Key challenges and techniques
Follow-ups, extensions, future directions

First Part

Contract Design

Incentivize agents to act in our interest through a payment contract.

Agents' actions are hidden (moral hazard)

Actions lead to observable outcomes

We wish to design an outcome-dependent contract

Contract Design

Incentivize agents to act in our interest through a payment contract.

Goal:
Incentivize a sales team
to sell a product

Hidden Action:
Thorough customer research

Incentive:
Commission per sale (2%)

Contract Design

Incentivize agents to act in our interest through a payment contract.

Goal:
Incentivize a team of programmers

Hidden Action:
Careful code review

Incentive:
Equity stake (0.5%)

Contract Design

Incentivize agents to act in our interest through a payment contract.

Goal:
Incentivize a national team
to win the World Cup

Hidden Action:
Intense training

Incentive:
Prize for winning ($500K)

Contract Design

Incentivize agents to act in our interest through a payment contract.

Defining Feature:
Payments must be conditional on observable outcomes,
because agent actions are hidden.

Challenge 1:
Outcomes are noisy signals since
the same outcome may arise from different actions.

Challenge 2:
Free riding is an issue since
individual contributions are unobservable.

Algorithmic Contract Design

Incentivize agents to act in our interest through a payment contract.

Contract design is increasingly moving online and growing in scale, mandating an algorithmic approach to contract theory.

Studying mechanism design through the algorithmic lens helped to:
(a) reveal structural insights
(b) identify tractability frontiers
(c) inform the design of simple mechanisms

A Renaissance of Contracts

1982 — **Moral Hazard in Teams (Bell J. Econ)** — **Holmström**

e.g. [Holmström, Milgrom, 1990] [Itoh, 1991]
[Legros, Matsushima, 1991]

2006 — **Combinatorial Agency (EC)** — **Babaioff, Feldman, Nisan**

e.g. [Babaioff, Feldman, Nisan, 2006 & 2009]
[Babaioff, Winter, 2014] [Emek, Feldman, 2014]

2016 — **Nobel Prize in Economics** — **Hart, Holmström**

e.g. [Ho, Slivkins, Vaughan, 2016]
[Dütting, Roughgarden, Talgam-Cohen, 2018 & 2020]

2023 — **Multi-Agent Contracts (STOC)**

2025 — **Multi-Agent Combinatorial Contracts (SODA)**

Dütting, Ezra, Feldman, Kesselheim

A Renaissance of Contracts

[Deo-Campo Vuong, Dughmi, Patel, Prasad, SODA 2024]
[Ezra, Feldman, Schlesinger, ITCS 2024]
[Alon, Castiglioni, Chen, Ezra, Li, Talgam-Cohen, EC 2025]
[Aharoni, Hoefer, Talgam-Cohen, EC 2025]
[Castiglioni, Chen, Li, 2025 & 2025]
[Feldman, Gal-Tzur, Ponitka, Schlesinger, EC 2025 & ITCS 2026 & EC 2026]
[Doron-Arad, Shachnai, Shmerler, Talgam-Cohen, EC 2026]
[Dütting, Ezra, Feldman, Kesselheim, EC 2026]
[Ding, Li, Sun, 2026]
[Lavi, Shachnai, Talgam-Cohen, 2026]

Excellent Related Resources

EC 2019 Tutorial

Contract Theory: A New Frontier for AGT

STOC 2022 Workshop

Optimizing the Effort of Others

EC 2022 Workshop

Algorithmic Contract Design: Present and Future

The above focus mainly on single-agent contracts.

Our tutorial is on multi-agent contracts.

Algorithmic Contract Theory: A Survey
[Dütting, Feldman, Talgam-Cohen '24]

Combinatorial Contract Design: Recent Progress and Emerging Frontiers
[Feldman '25]

Contract Design

Incentivize agents to act in our interest through a payment contract.

Goal:
Incentivize a sales team
to sell a product

Hidden Action:
Thorough customer research

Incentive:
Commission per sale (2%)

Contract Design

Principal

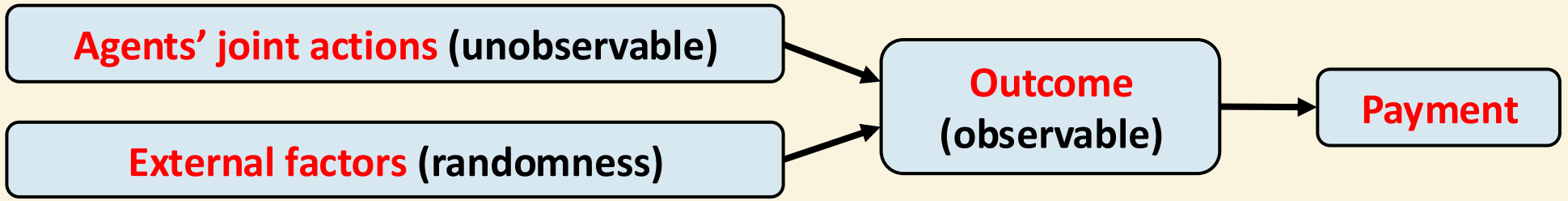
owns a binary-outcome project,
receives a reward if it succeeds.

Agents




affect the probability of success
through their costly actions.

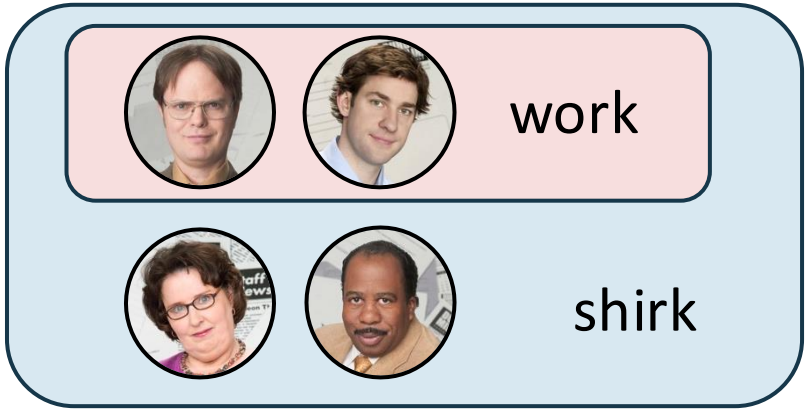
A contract incentivizes the agents to take actions
by offering each agent a payment conditional on success.



Action Models

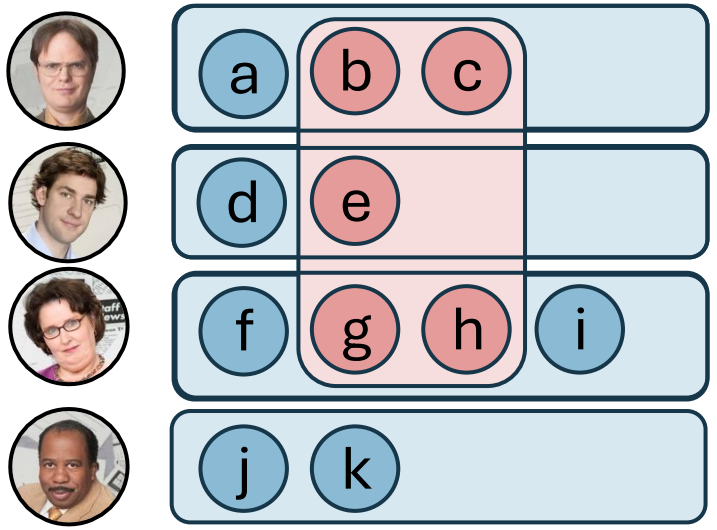
Binary-Action Model
(First Part)

Multi-Agent Contracts
(DEFK ‘23)



Multi-Action Model
(Second Part)

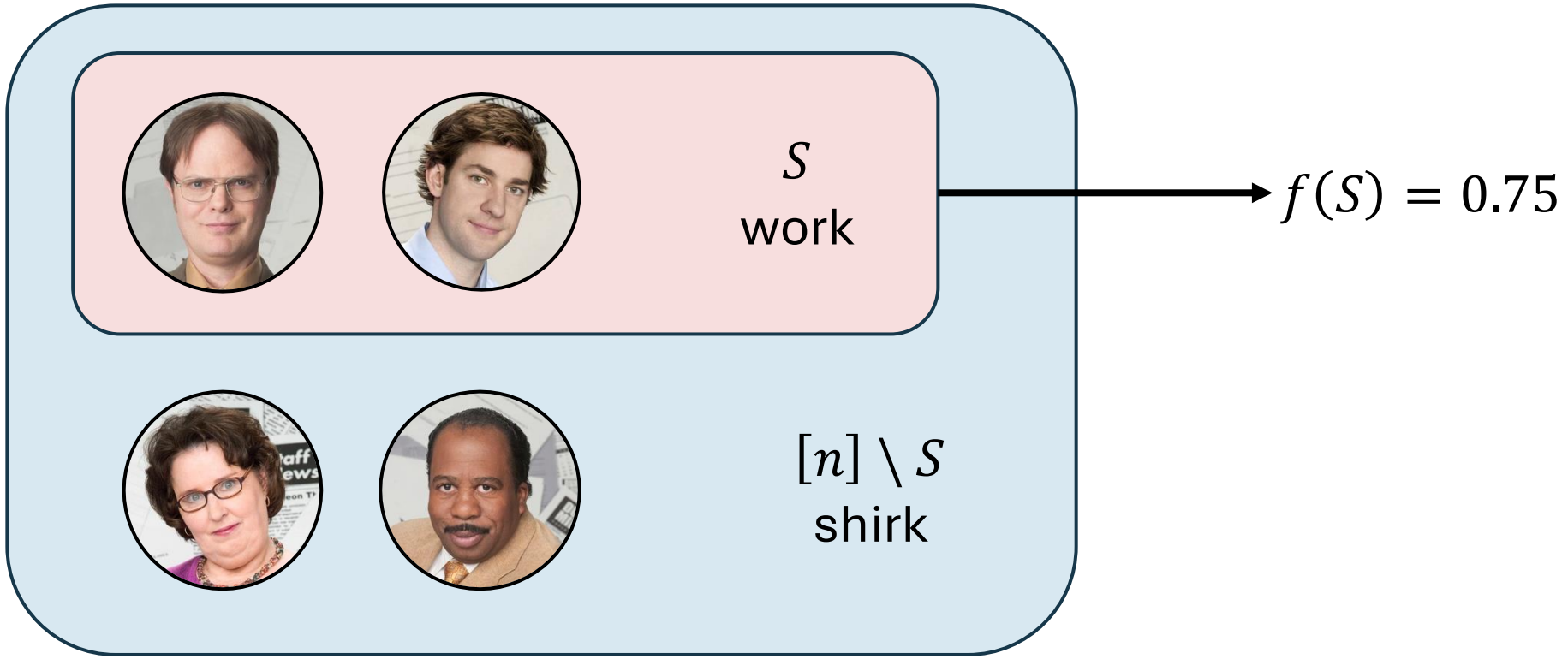
Multi-Agent Combinatorial Contracts
(DEFK ‘25)



Binary-Action Model

Multi-Agent Binary-Action Model [DEFK '23]

Known: n agents. Each agent i either works (at cost c_i) or shirks (at cost 0).
Function $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ maps set of working agents S to success prob. $f(S)$.



Multi-Agent Binary-Action Model [DEFK '23]

Known: n agents. Each agent i either works (at cost c_i) or shirks (at cost 0).
Function $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ maps set of working agents S to success prob. $f(S)$.

Step 1. Principal commits to a **contract** $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n) \in [0,1]^n$
offering agent i an α_i-fraction of the reward.


$\alpha_1 = 0.1$


$\alpha_2 = 0.2$


$\alpha_3 = 0$


$\alpha_4 = 0$

For binary outcomes, such *linear contracts* are without loss of generality.

Multi-Agent Binary-Action Model [DEFK '23]

Known: n agents. Each agent i either works (at cost c_i) or shirks (at cost 0).
Function $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ maps set of working agents S to success prob. $f(S)$.

Step 1. Principal commits to a **contract** $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n) \in [0,1]^n$
offering agent i an α_i-fraction of the reward.


$\alpha_1 = 0.1$


$\alpha_2 = 0.2$


$\alpha_3 = 0$

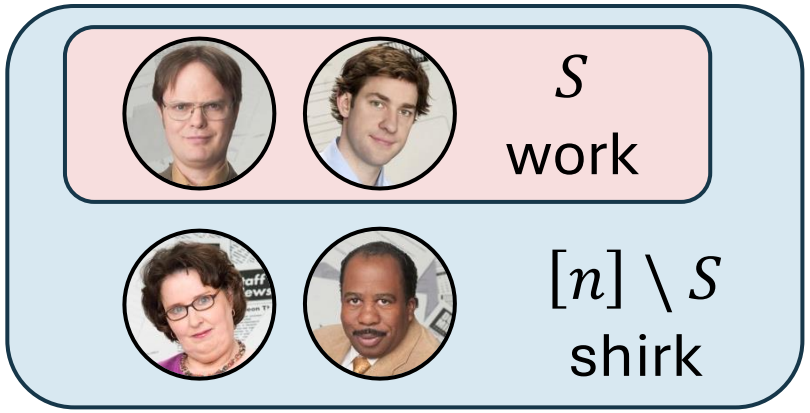

$\alpha_4 = 0$

Multi-Agent Binary-Action Model [DEFK '23]

Known: n agents. Each agent i either works (at cost c_i) or shirks (at cost 0).
Function $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ maps set of working agents S to success prob. $f(S)$.

Step 1. Principal commits to a **contract** $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n) \in [0,1]^n$
offering agent i an α_i-fraction of the reward.

Step 2. Each agent i strategically chooses to work or shirk.
Agents form a **pure Nash equlibrium** $S \subseteq [n]$; ties broken to maximize $f(S)$.



Multi-Agent Binary-Action Model [DEFK '23]

Known: n agents. Each agent i either works (at cost c_i) or shirks (at cost 0).
Function $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ maps set of working agents S to success prob. $f(S)$.

Step 1. Principal commits to a **contract** $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n) \in [0,1]^n$
offering agent i an α_i-fraction of the reward.

Step 2. Each agent i strategically chooses to work or shirk.
Agents form a **pure Nash equlibrium** $S \subseteq [n]$; ties broken to maximize $f(S)$.

Step 3: Outcome is realized: either success (reward 1) or failure (reward 0).

Agent i's utility:
$\alpha_i \cdot f(S) - c_i \cdot \mathbf{1}[i \in S]$

Principal's utility (profit):
$(1 - \sum_{i=1}^{n} \alpha_i) \cdot f(S)$

Multi-Agent Binary-Action Model

Our Goal: efficiently compute contract $\vec{\alpha} \in [0,1]^n$ and $S \subseteq [n]$ which approx. maximizes principal's utility $(1 - \sum_i \alpha_i) f(S)$
Subject to S being a Nash equlibrium induced by $\vec{\alpha}$.

remaining reward **success probability**

maximize $(1 - \sum_{i=1}^{n} \alpha_i) \cdot f(S)$

subject to

$\alpha_i \cdot f(S) - c_i \geq \alpha_i \cdot f(S \setminus \{i\})$ $\forall i \in S$

$\alpha_i \cdot f(S) \geq \alpha_i \cdot f(S \cup \{i\}) - c_i$ $\forall i \in [n] \setminus S$


$\alpha_1 = 0.1$


$\alpha_2 = 0.2$

S
work


$\alpha_3 = 0$


$\alpha_4 = 0$

$[n] \setminus S$
shirk

All About f

This hierarchy of set functions captures the combinatorial structure of $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$.

XOS (fractionally subadditive)

$$f(S) = \max_{\ell \in L} \ell(S)$$

L is a collection of additive functions

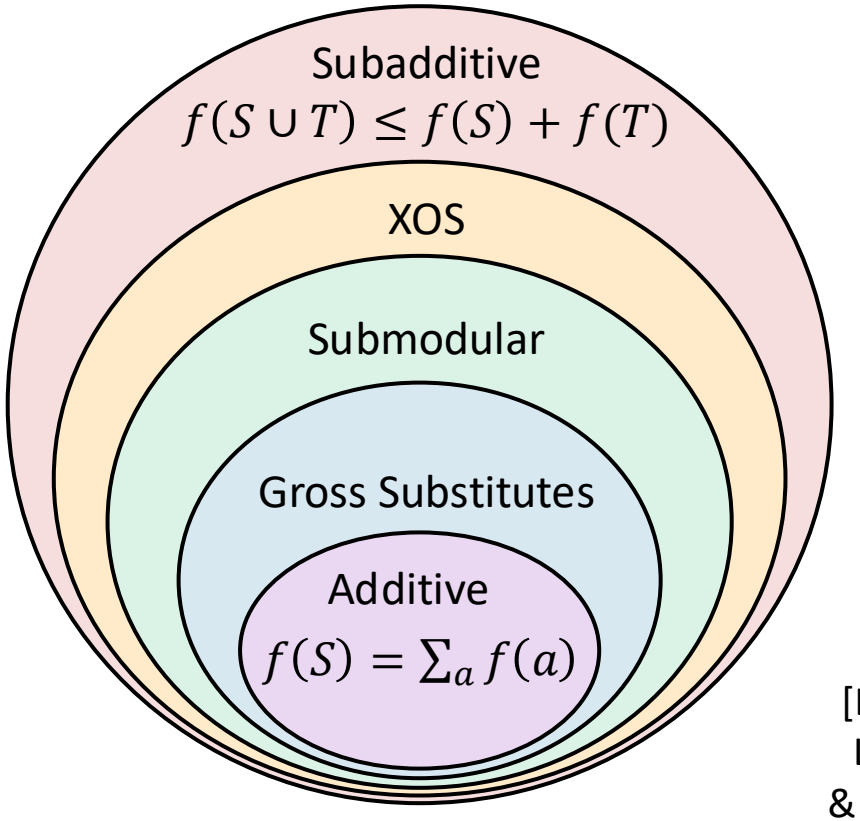
(marginals can decrease or increase)

Submodular f

$$\forall\, S \subseteq T \subseteq A, \qquad \forall i \in S$$

$f_i(S) \geq f_i(T)$ *(decreasing marginals)*

$$f_i(S) = f(S) - f(S \setminus \{i\})$$



[Lehmann, Lehmann & Nisan '01]

All About f

The function $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ has exponential representation size.
We access f via either:

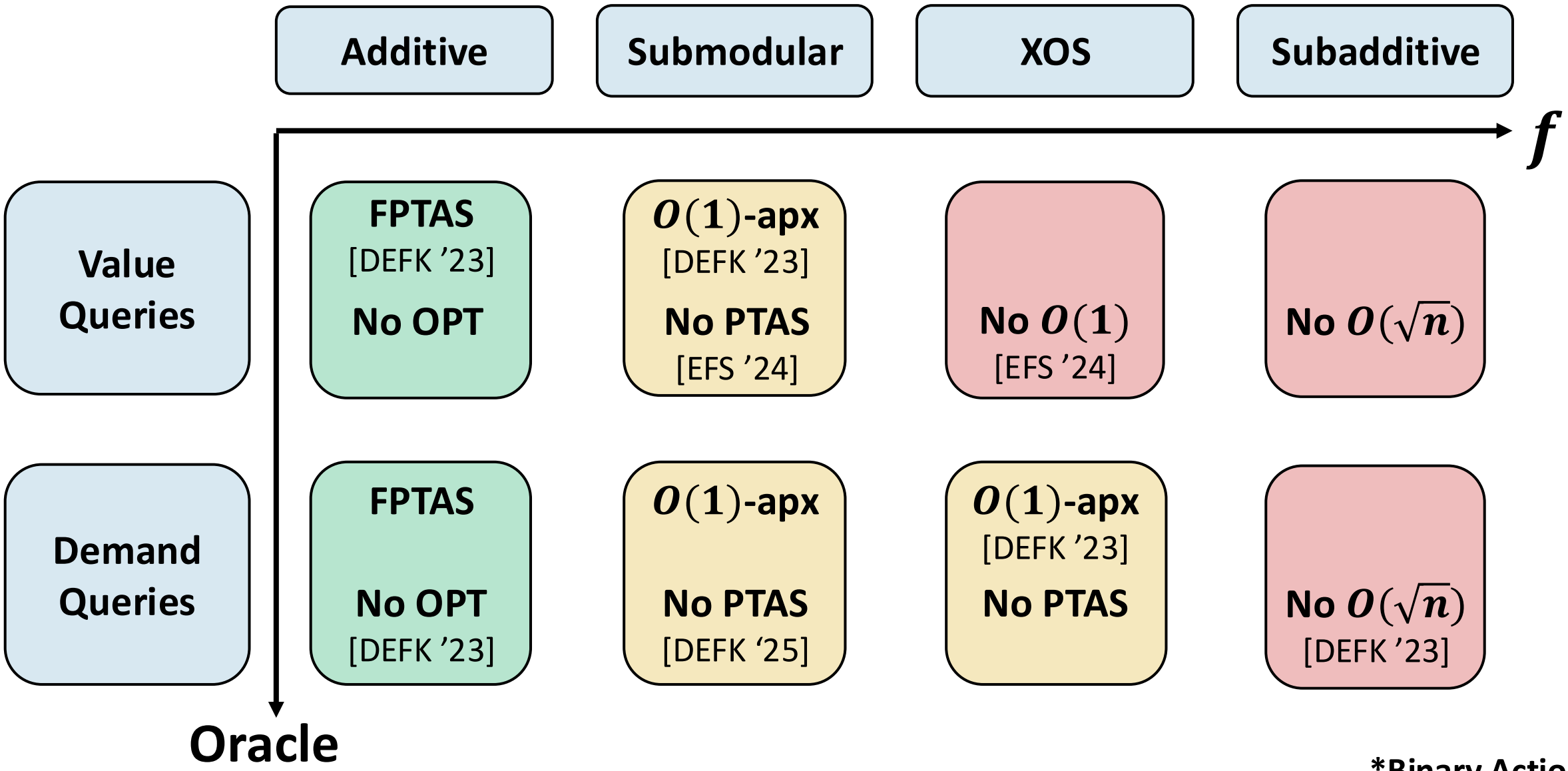
Value Queries
Input: $S \subseteq [n]$
Output: $f(S)$

Demand Queries
Input: $p \in \mathbb{R}^n$
Output: $D \in \underset{S \subseteq [n]}{\operatorname{argmax}} \left(f(S) - \sum_{a \in S} p_a\right)$

Value queries can be approximately simulated with demand queries.

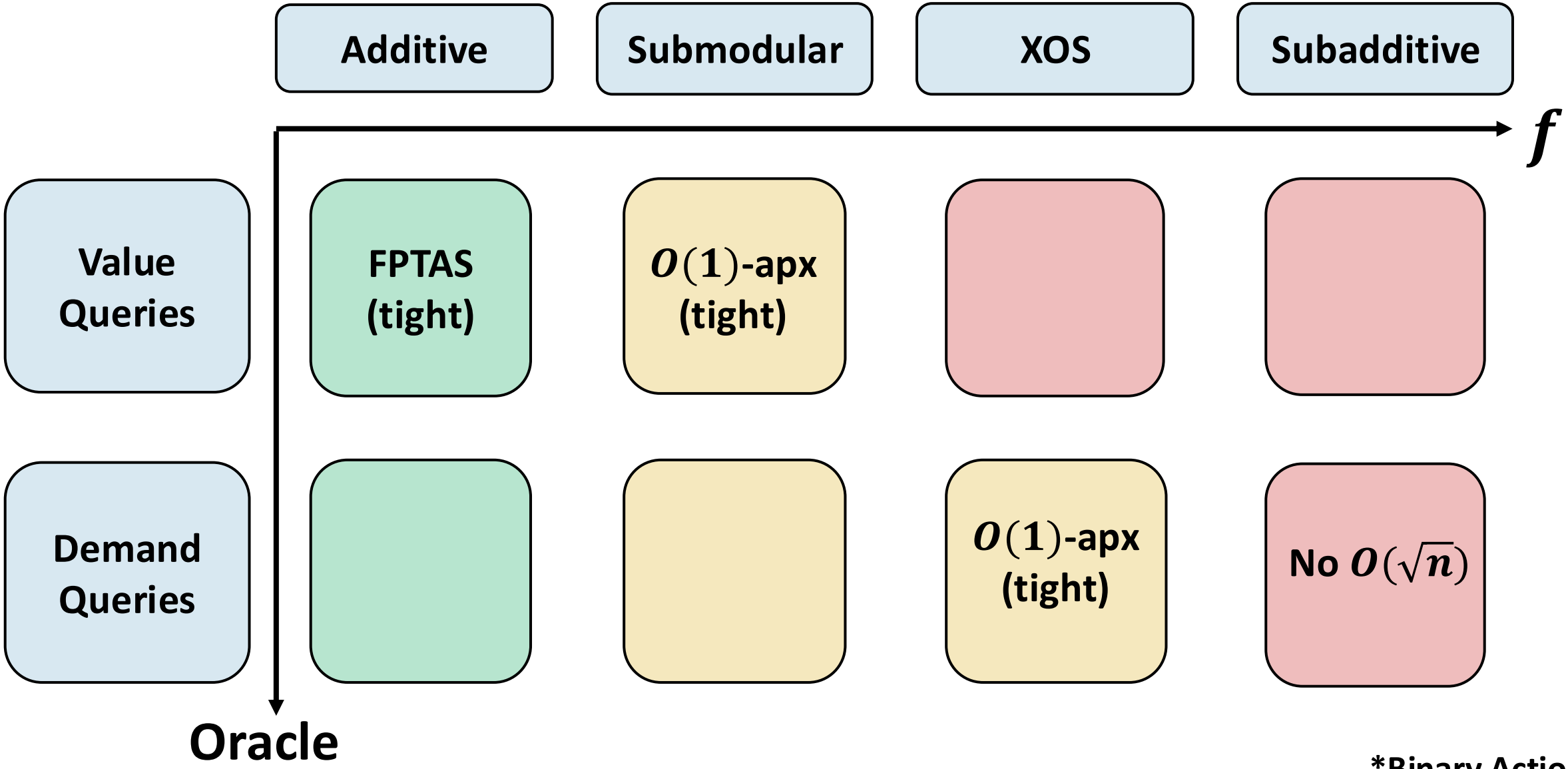
Results for Binary Actions

The Computational Landscape*



*Binary Actions

The Computational Landscape*



*Binary Actions

Challenges and Techniques for Binary Actions

Simplifying the Optimization Problem

$$\max_{\alpha,S} \left(1 - \sum_{i=1}^{n} \alpha_i\right) \cdot f(S) \quad s.t.$$

$$\alpha_i \cdot f(S) - c_i \geq \alpha_i \cdot f(S \setminus \{i\}) \quad \forall i \in S$$

$$\alpha_i \cdot f(S) \geq \alpha_i \cdot f(S \cup \{i\}) - c_i \quad \forall i \notin S$$

$$\max_{S} g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f_i(S)}\right) \cdot f(S)$$

where $f_i(S) = f(S) - f(S \setminus \{i\})$
(marginal contribution to S)

Optimal Payments: For every agent $i \notin S$, set $\alpha_i = 0$.
For every agent $i \in S$, the following must hold

$$\alpha_i \cdot f(S) - c_i \geq \alpha_i \cdot f(S \setminus \{i\}) \quad \Leftrightarrow \quad \alpha_i \geq \frac{c_i}{f_i(S)}$$

utility for exerting effort

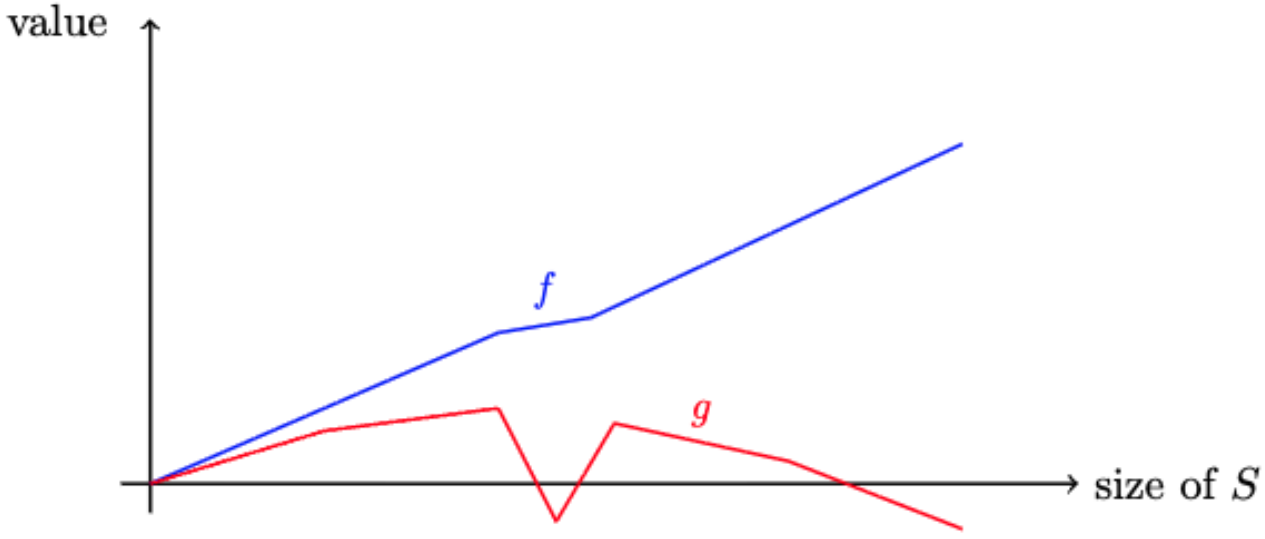
utility for shirking

Objective g Lacks Structure

$$g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f_i(S)}\right) \cdot f(S)$$

Example: $f(S)$ depends only on $|S|$, and f is XOS

g is a mixed-sign objective:
not monotone and possibly negative



additive f	$\Longrightarrow$	submodular g
submodular f	$\Longrightarrow$	subadditive g
XOS f	$\Longrightarrow$	no guarantees

Verify additive to SM

Three Recurring Techniques

Goal: maximize $g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f_i(S)}\right) \cdot f(S)$

Technique 1. Decomposing g (AKA light/heavy agents)
[DEFK '23, DEFK '25, FGPS '25, FGPS '26, DFGK '26, ACC+ '25]

Technique 2. How to query a demand oracle
[DEFK '23, DEFK '25, FGPS '26, DFGK '26]

Technique 3. Hide a special set of agents
[DEFK '23, DDPP '24, DEFK '25, EFS '24, FGPS '26]

Decomposing g

Let $S^\star \in \operatorname{argmax}_S \left(g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f_i(S)}\right) \cdot f(S) \right)$

Goal: Find $S \subseteq [n]$ such that $g(S) \geq \Omega(1) \cdot g(S^\star)$

For subadditive f, one of the following achieves this:

Best single agent

$$\max_i g(\{i\}) = \left(1 - \frac{c_i}{f_i}\right) f_i = f_i - c_i$$

(Naively in poly-time)

Best team under budget

$$\max_S f(S) \text{ s.t. } \sum_{i \in S} \frac{c_i}{f_i(S)} \leq 1/2$$

(The hard part)

How to Query a Demand Oracle?

Goal: Maximize $f(S)$ subject to $\sum_{i \in S} \frac{c_i}{f_i(S)} \leq 1/2$

Demand Query. $p \in \mathbb{R}^{[n]} \mapsto D \in \arg\max_{S \subseteq [n]} (f(S) - \sum_{i \in S} p_i)$

Lemma 1 [DEFK '23]. Let $S^\star \in \arg\max g(S)$. It holds that $\sum_{i \in S^\star} \sqrt{c_i} \leq \sqrt{f(S^\star)}$

Run demand query with prices $p_i = \frac{1}{2}\sqrt{c_i f(S^\star)}$. Let D be a demand set.

$$f(D) \geq f(S^\star) - \sum_{i \in S^\star} p_i = f(S^\star) - \frac{1}{2}\sqrt{f(S^\star)} \sum_{i \in S^\star} \sqrt{c_i} \geq \frac{1}{2} f(S^\star)$$

Assume for now that $f(S^\star)$ is known

Solving the Budgeted Problem (Almost)

Goal: Maximize $f(S)$ subject to $\sum_{i \in S} \frac{c_i}{f_i(S)} \leq 1/2$

Run demand query with prices $p_i = \frac{1}{2}\sqrt{c_i f(S^\star)}$.

This yields a rewarding set D with $f(D) \geq f(S^\star)/2$

What about the budget constraints? One can efficiently compute $D' \subseteq D$

1. Reward guarantee: $f(D') \geq \frac{f(D)}{O(1)} \geq \frac{f(S^\star)}{2 \cdot O(1)}$
2. Budget feasibility: Payments at most $1/2$

$O(1)$-Approx. to the Principal's Utility

Goal: Find $S \subseteq [n]$ such that $g(S) \geq \Omega(1) \cdot g(S^\star)$

Best single agent
(Brute force)

Best team with payment $\leq 1/2$
(The hard part)

Repeat and guess $f(S^\star)$

$$D \in \arg\max f(S) - \sum_{i \in S} \frac{1}{2}\sqrt{c_i f(S^\star)}$$

Compute a good subset $D' \subseteq D$

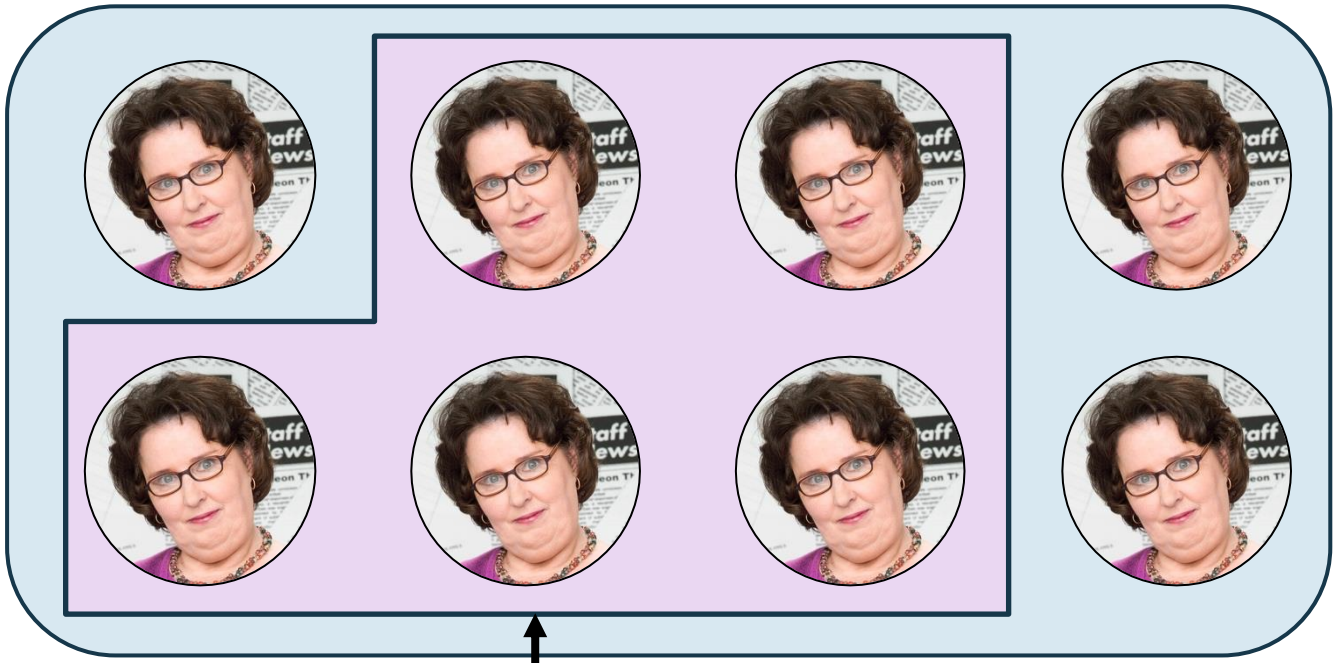
Hide a Special Set

(a) A hidden set is required to find a good contract.
(b) Exponentially-many queries required to find the hidden set.

(Next slide) Theorem [DEFK '23]:
No $O(\sqrt{n})$-approximation
for subadditive f.

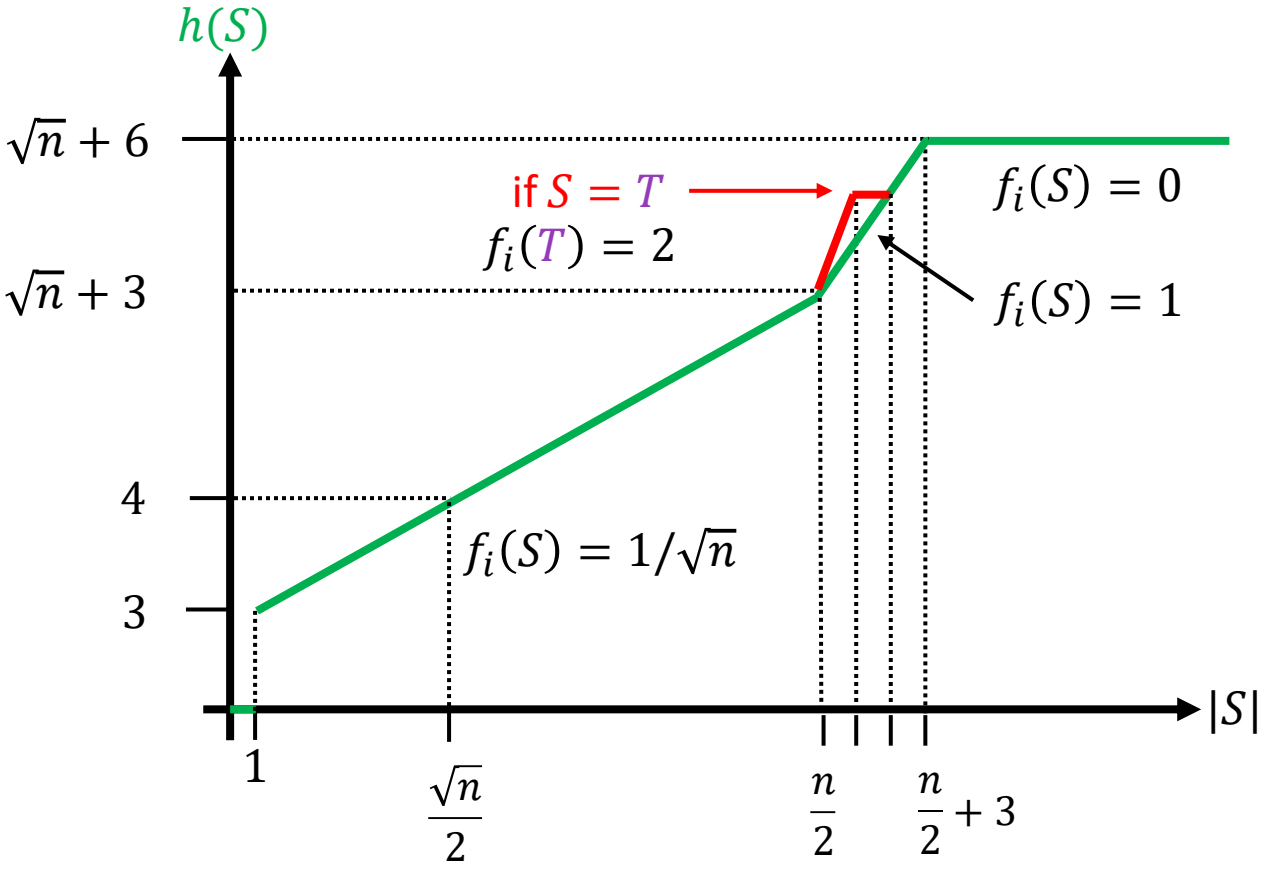
Many other hardness results
rely on this technique.
[DEFK '25, EFS '24, FGPS '26]

Unconditional hardness



hidden set of agents T

Hide a Special Set for Subadditive f



For T of size $n/2+1$,
$f(S)=h(S)+\mathbb{1}[S=T]$

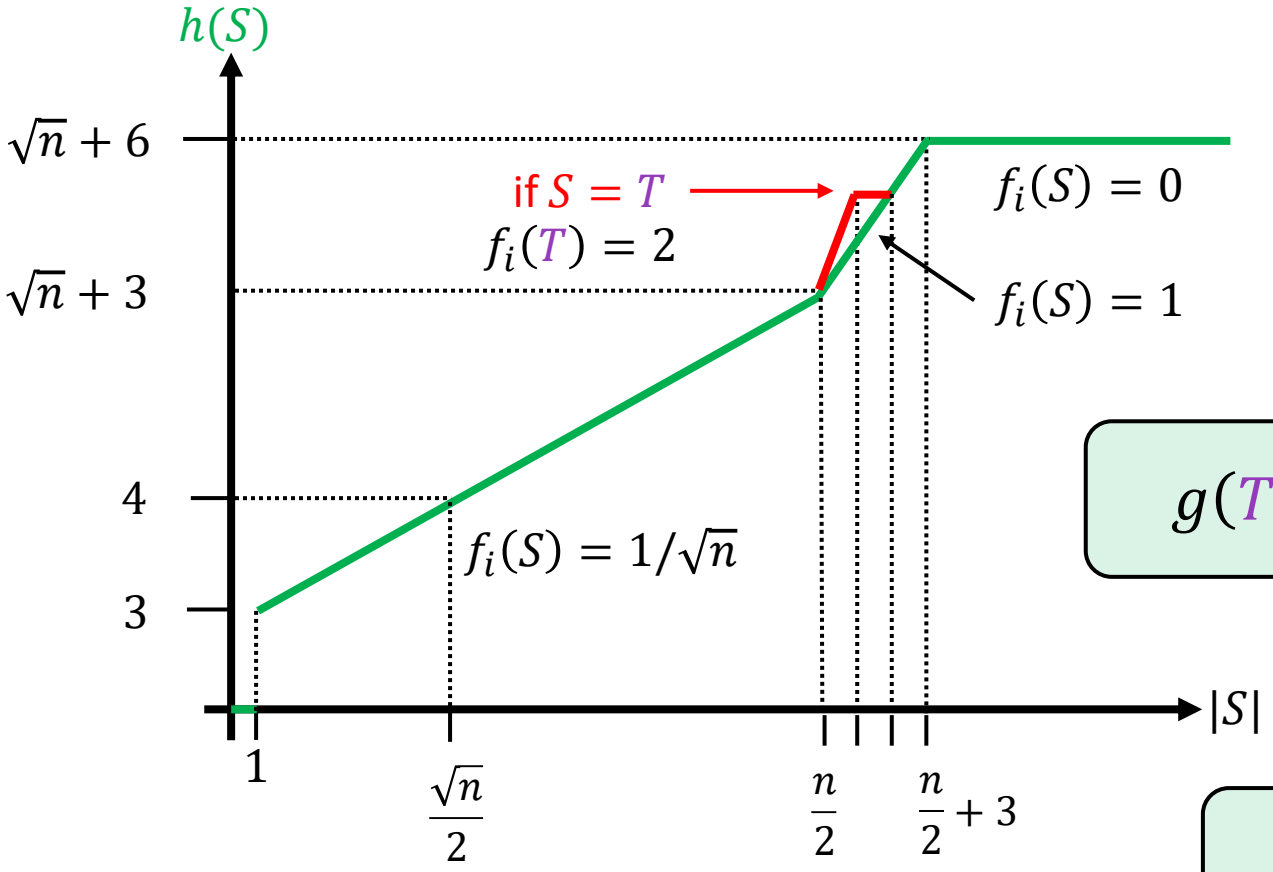
Set $c_i = 2/n$.

Recall $g(S)=\left(1-\sum_{i\in S}\frac{c_i}{f_i(S)}\right)\cdot f(S)$

For $|S| \le n/2$, payments ≤ 1 imply $|S| \le \sqrt{n}$

For $|S| \ge \frac{n}{2}+3$, marginals are 0. Agents with 0 marginal cannot be incentivized.

Hide a Special Set for Subadditive f



Set $c_i = 2/n$.

Recall $g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f_i(S)}\right) \cdot f(S)$

$$g(T) = \left(1 - \left(\frac{n}{2}+1\right) \cdot \frac{2/n}{2}\right) \cdot \Omega(\sqrt{n}) = \Omega(\sqrt{n})$$

$|T|$ $\quad \frac{c_i}{f_i(T)}$ $\quad f(T)$

$$g(S) \le \left(1 - \left(\frac{n}{2}\right) \cdot \frac{2/n}{1}\right) \cdot \Omega(\sqrt{n}) \le 0 \qquad (S \neq T)$$

For T of size $n/2+1$,
$f(S) = h(S) + \mathbb{1}[S = T]$

See [DEFK '23]: (a) Demand queries reveal negligible information on T. (b) f is monotone subadditive.

Second Part

Part II Outline

The multi-agent multi-action model

New challenges and techniques

Extensions (of binary-action and multi-action models):

Budgets and fairness constraints
Project assignment
Beyond principal's utility

Future directions

Multi-Action Model

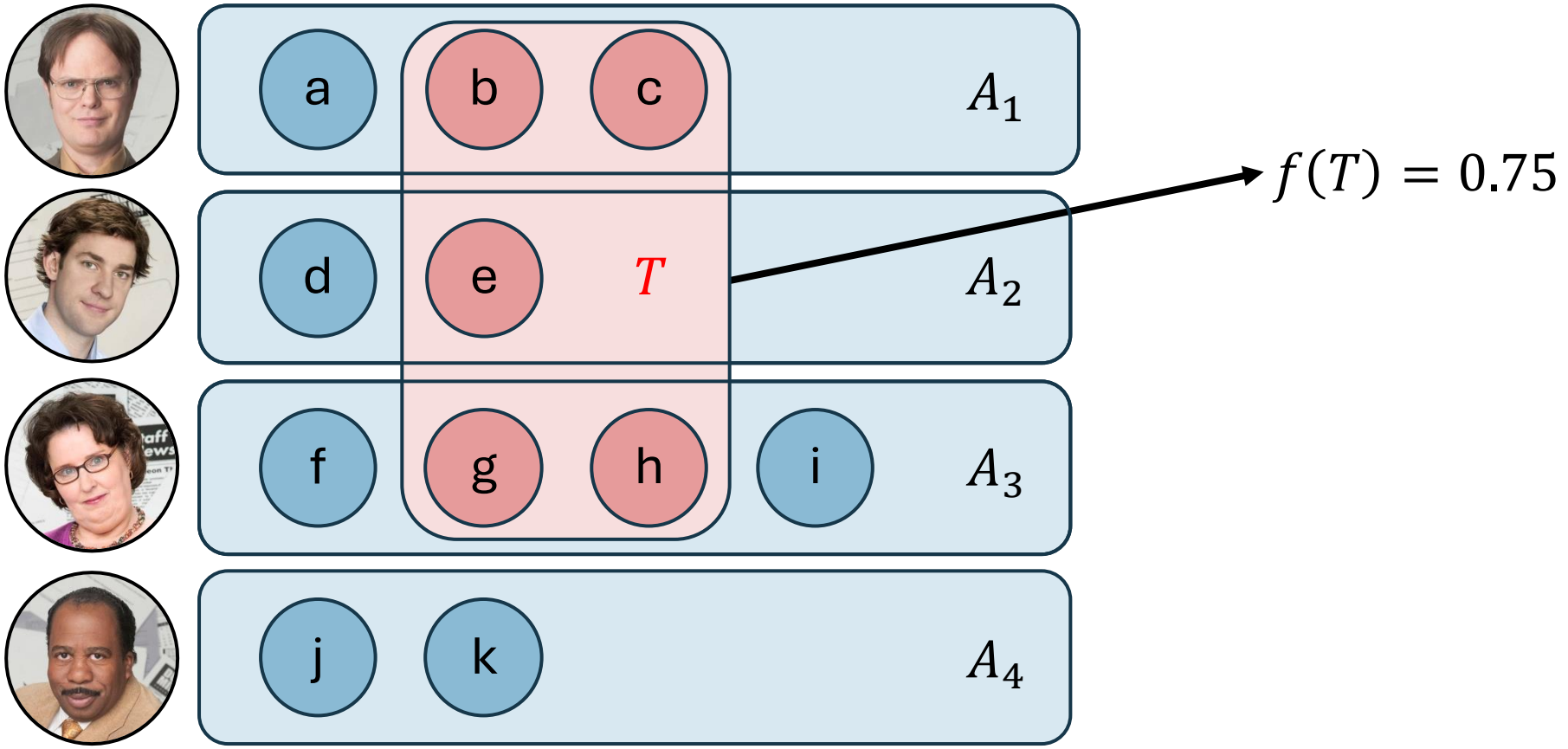
Multi-Agent Multi-Action Model

Known: n agents. Each agent i has set of individual actions A_i.
A is the union of all actions.

If $|A_i| = 1$ for all i, then we are in the binary-action model [DEFK '23]

Multi-Agent Multi-Action Model

Known: n agents. Each agent i has set of individual actions A_i.
Function $f : 2^A \to \mathbb{R}_{\geq 0}$ maps set of taken actions T to success prob. $f(T)$.



Multi-Agent Multi-Action Model

Known: n agents. Each agent i has set of individual actions A_i.
Function $f : 2^A \to \mathbb{R}_{\geq 0}$ maps set of taken actions T to success prob. $f(T)$.

Step 1. Principal commits to a contract $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n) \in [0, 1]^n$
offering agent i an α_i-fraction of the reward.


$\alpha_1 = 0.1$


$\alpha_2 = 0.2$


$\alpha_3 = 0.3$

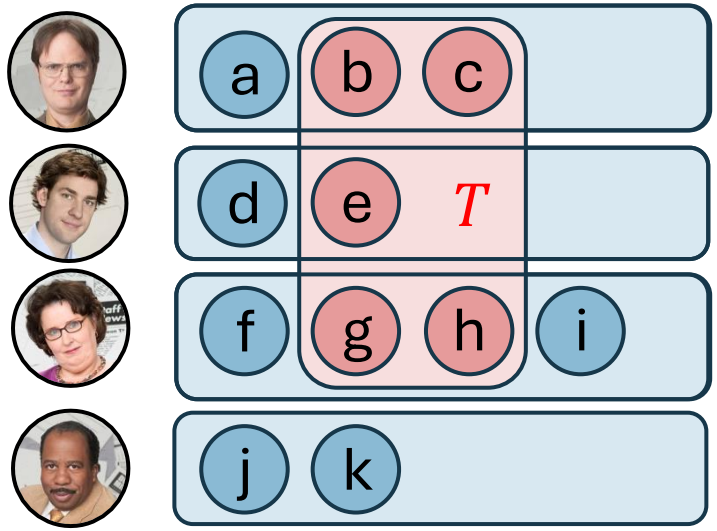

$\alpha_4 = 0$

Multi-Agent Multi-Action Model

Known: n agents. Each agent i has set of individual actions A_i.
Function $f : 2^A \to \mathbb{R}_{\geq 0}$ maps set of taken actions T to success prob. $f(T)$.

Step 1. Principal commits to a contract $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n) \in [0, 1]^n$
offering agent i an α_i-fraction of the reward.

Step 2. Each agent i strategically chooses a subset of actions to take.
Agents form a pure Nash equlibrium $T \subseteq A$; ties broken to maximize $f(T)$.



Multi-Agent Multi-Action Model

Known: n agents. Each agent i has set of individual actions A_i.
Function $f : 2^A \to \mathbb{R}_{\geq 0}$ maps set of taken actions T to success prob. $f(T)$.

Step 1. Principal commits to a contract $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n) \in [0,1]^n$ offering agent i an α_i-fraction of the reward.

Step 2. Each agent i strategically chooses a subset of actions to take.
Agents form a pure Nash equlibrium $T \subseteq A$; ties broken to maximize $f(T)$.

Step 3. Outcome is realized: either success (reward 1) or failure (reward 0).

Agent i's utility:
$\alpha_i \cdot f(T) - \sum_{a \in T_i} c_a$

Principal's utility (profit):
$(1 - \sum_{i=1}^{n} \alpha_i) \cdot f(T)$

Multi-Agent Multi-Action Model

Our Goal: efficiently compute contract $\vec{\alpha} \in [0,1]^n$ and $T \subseteq A$ which approximately maximizes principal's utility subject to T being a Nash equlibrium with respect to $\vec{\alpha}$.

maximize $\left(1 - \sum_{i=1}^{n} \alpha_i\right) \cdot f(T)$

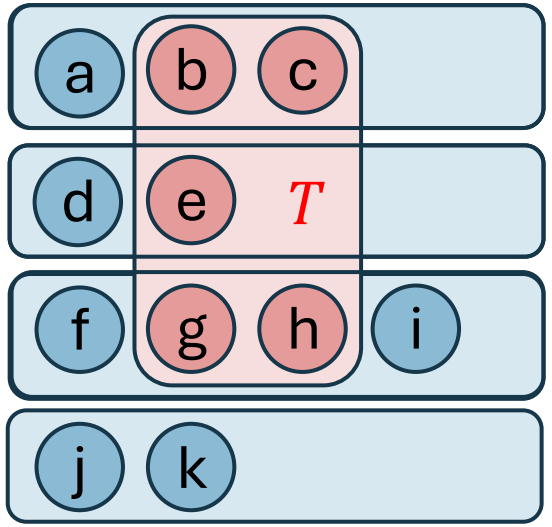
subject to $\alpha_i \cdot f(T) - \sum_{a \in T_i} c_a \geq \alpha_i \cdot f(T_i' \cup T_{-i}) - \sum_{a \in T_i'} c_a \quad \forall\, i \;\; \forall\, T_i' \subseteq A_i$

$\alpha_1 = 0.1$ 

$\alpha_2 = 0.2$ 

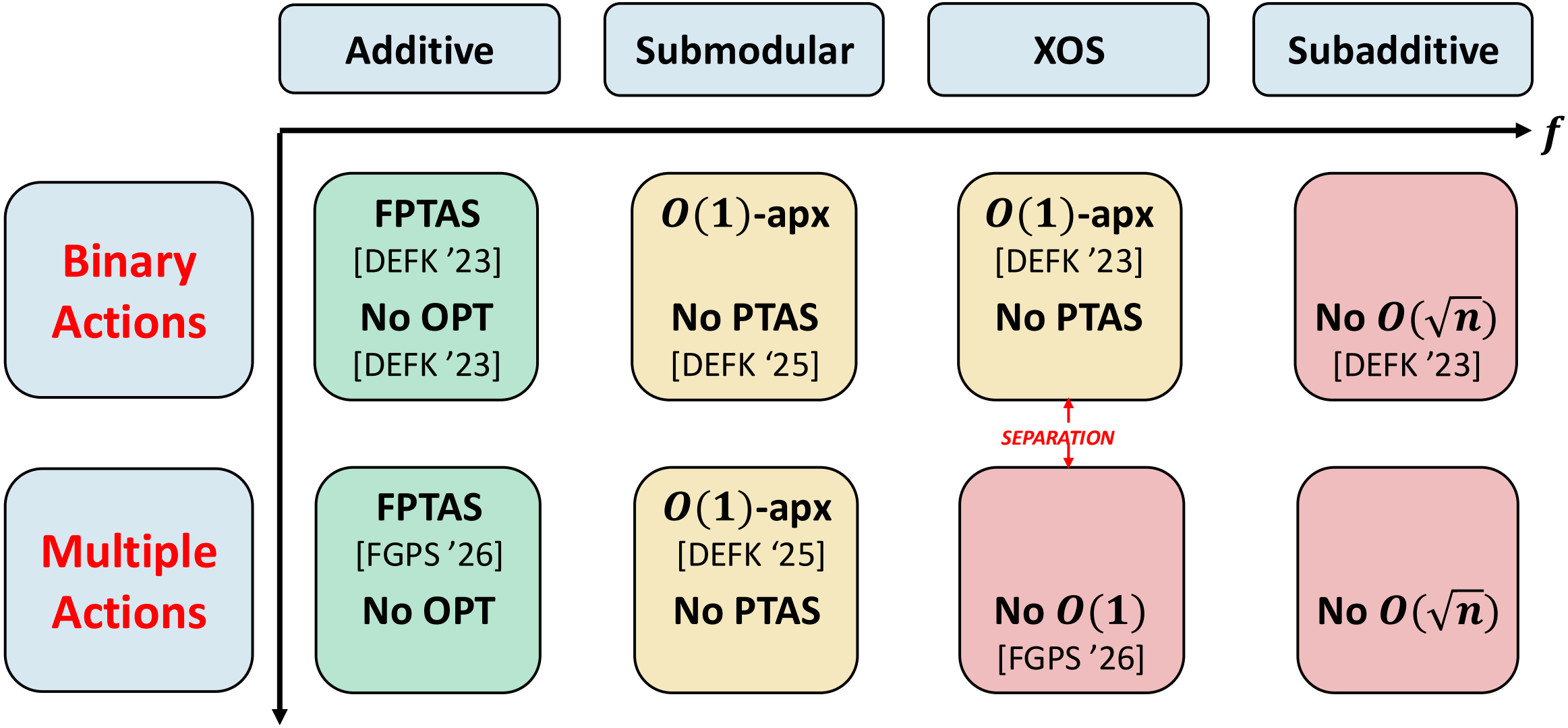
$\alpha_3 = 0.3$ 

$\alpha_4 = 0$ 



Results for Multi-Action

The Computational Landscape*

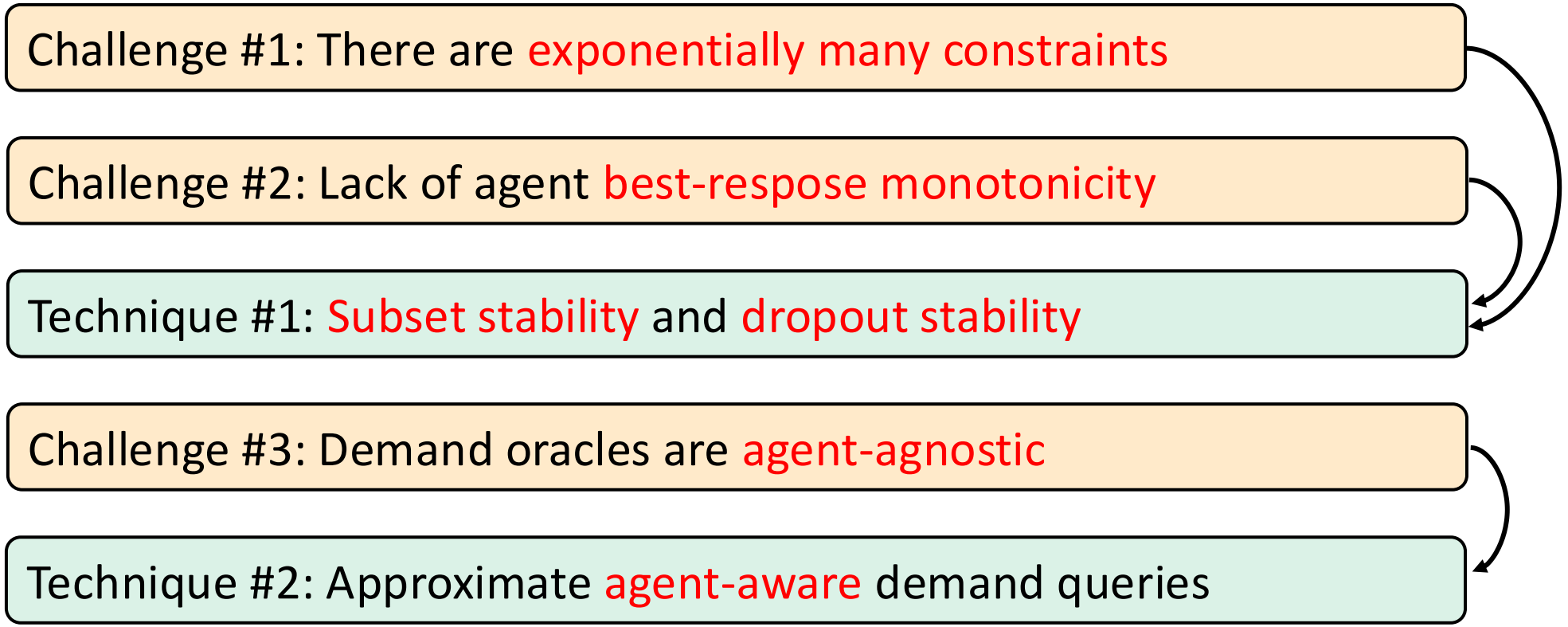


*Assuming Value and Demand Queries

New Challenges and Techniques

Key Challenges and Techniques

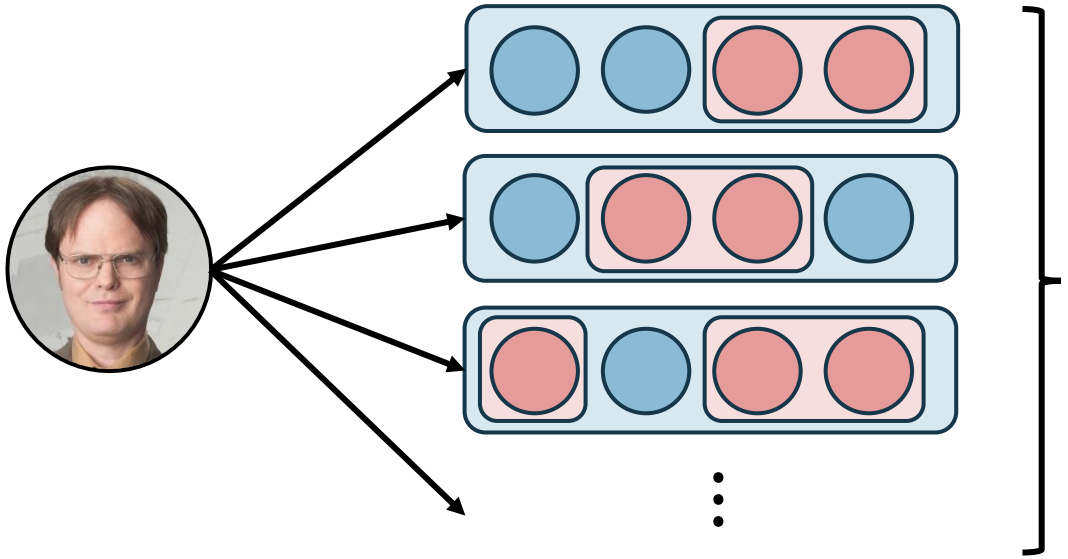
Recurring themes behind the positive and negative results for the multi-action model



Exponentially Many Constraints

maximize $\left(1 - \sum_{i=1}^{n} \alpha_i\right) \cdot f(T)$

subject to $\alpha_i \cdot f(T) - \sum_{a \in T_i} c_a \geq \alpha_i \cdot f(T_i' \cup T_{-i}) - \sum_{a \in T_i'} c_a \quad \forall\, i \;\; \forall\, T_i' \subseteq A_i$



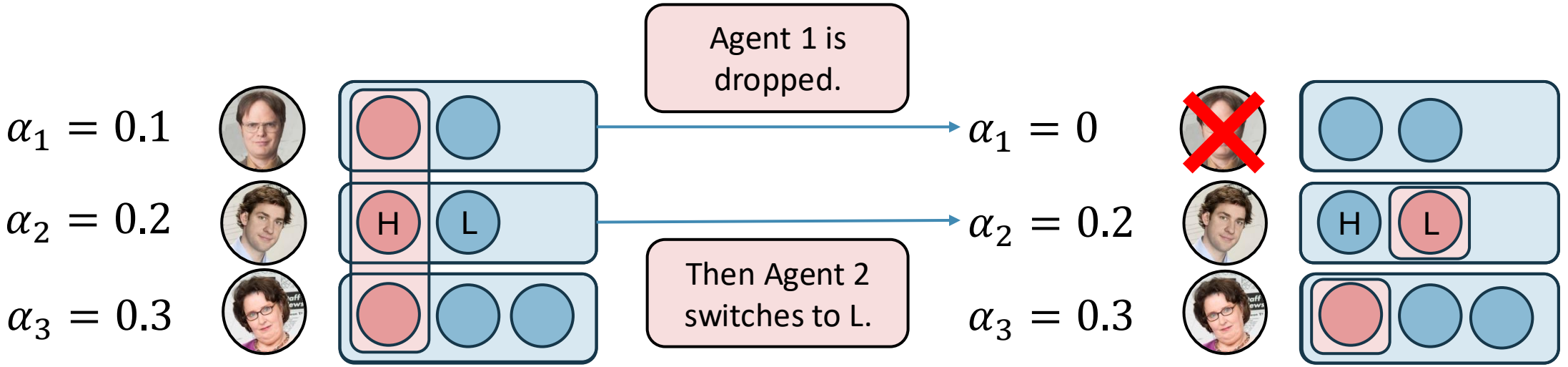
Exponentially-many possible deviations!

No closed-form payment rule, unlike for binary actions, where the optimal payment is $\alpha_i = \frac{c_i}{f_i(S)}$.

Lack of Best-Response Monotonicty

Best-Response Monotonicity: When an agent is dropped, each remaining agent will only add actions.

Can be violated for multiple actions with submodular f.



Holds for binary actions with submodular f, since threshold for exerting effort is $\frac{c_i}{f_i(S)} \geq \frac{c_i}{f_i(S \setminus \{j\})}$ by submodularity of f.

Subset Stability and Dropout Stability

Pure Nash Equilibrium (PNE)

maximize $(1 - \sum_{i=1}^{n} \alpha_i) \cdot f(T)$

subject to $\alpha_i \cdot f(T) - \sum_{a \in T_i} c_a \geq \alpha_i \cdot f(T_i' \cup T_{-i}) - \sum_{a \in T_i'} c_a \quad \forall\, i \quad \forall\, T_i' \subseteq A_i$

Relaxed Constraint #1: Subset Stability [DEFK '25]

$\alpha_i \cdot f(T) - \sum_{a \in T_i} c_a \geq \alpha_i \cdot f(T_i' \cup T_{-i}) - \sum_{a \in T_i'} c_a \quad \forall\, i \quad \forall\, T_i' \subseteq T_i$

Relaxed Constraint #2: Dropout Stability [DEFK '26]

$\alpha_i \cdot f(T) - \sum_{a \in T_i} c_a \geq \alpha_i \cdot f(T_i' \cup T_{-i}) - \sum_{a \in T_i'} c_a \quad \forall\, i \quad (T_i' = \emptyset)$

Doubling Lemmas: Fix contract $\vec{\alpha}$.

For submodular f and subset stable T, **any** PNE T' of $2\vec{\alpha} + \vec{\varepsilon}$ satisfies $f(T') \geq f(T)/2$.

For XOS f and dropout stable T, **some** PNE T' of $2\vec{\alpha}$ satisfies $f(T') \geq f(T)/2$.

Demand Oracles Are Agent-Agnostic

maximize $\left(1 - \sum_{i=1}^{n} \alpha_i\right) \cdot f(T)$

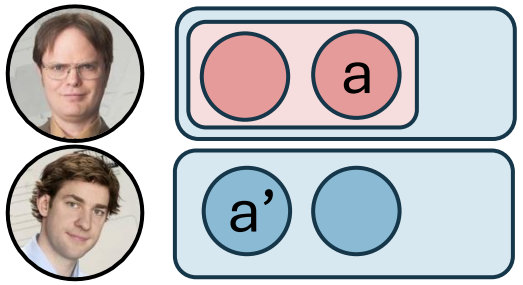
subject to $\alpha_i \cdot f(T) - \sum_{a \in T_i} c_a \geq \alpha_i \cdot f(T_i' \cup T_{-i}) - \sum_{a \in T_i'} c_a \quad \forall\, i \;\; \forall\, T_i' \subseteq A_i$

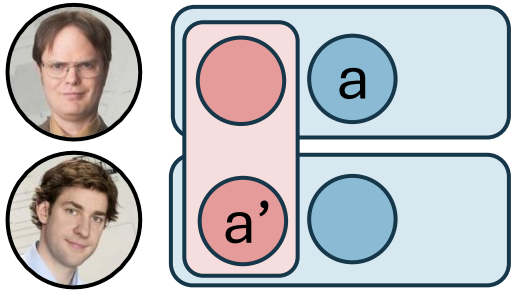
Demand Queries

$$\vec{p} \in \mathbb{R}^A \mapsto S \in \underset{S \subseteq A}{\operatorname{argmax}} \left(f(S) - \sum_{a \in S} p_a\right)$$

Demand queries are agent-agnostic:
They ignore ownership of actions

Payments are generally lower when fewer agents control the actions





Approximate Agent–Aware Demand Queries

maximize $\left(1 - \sum_{i=1}^{n} \alpha_i\right) \cdot f(T)$

subject to $\alpha_i \cdot f(T) - \sum_{a \in T_i} c_a \geq \alpha_i \cdot f(T_i' \cup T_{-i}) - \sum_{a \in T_i'} c_a \quad \forall\, i \;\; \forall\, T_i' \subseteq A_i$

Demand Queries

$\vec{p} \in \mathbb{R}^A \mapsto S \in \underset{S \subseteq A}{\operatorname{argmax}} \left(f(S) - \sum_{a \in S} p_a\right)$

← Agent-Agnostic

Idea #1: Bundle Pricing [DEFK '25]

$S \in \underset{S \subseteq T}{\operatorname{argmax}} \left(f(S) - \sum_i \sqrt{\sum_{a \in S_i} p_a}\right)$

Idea #2: Agent Cardinality Constraint [FGPS '26]

$S \in \underset{\#\text{agents}(S) \leq k}{\operatorname{argmax}} \left(f(S) - \sum_{a \in S} p_a\right)$

We can approximately solve both types of demand queries:

$f(S) - p(S) \geq \beta \cdot f(S^\star) - p(S^\star)$

Extensions

Extensions

Budget Constraints + Alternative Objectives
[Aharoni, Hoefer, Talgam-Cohen, EC 2025]
[Feldman, Gal-Tzur, Ponitka, Schlesinger, EC 2025 & ITCS 2026]

Supermodular Rewards
[Deo-Campo Vuong, Dughmi, Patel, Prasad, SODA 2024]

Multiple Projects
[Alon, Castiglioni, Chen, Ezra, Li, Talgam-Cohen, EC 2025]

Equal-Pay Contracts
[Feldman, Gal-Tzur, Ponitka, Schlesinger, EC 2026]

Budget-Feasible Contracts

Budget constraint: $\sum_{i\in[n]} \alpha_i \leq B$ for some $B \in [0,1]$.

[Aharoni, Hoefer, Talgam-Cohen, EC 2025]
[Feldman, Gal-Tzur, Ponitka, Schlesinger, EC 2025 & ITCS 2026]

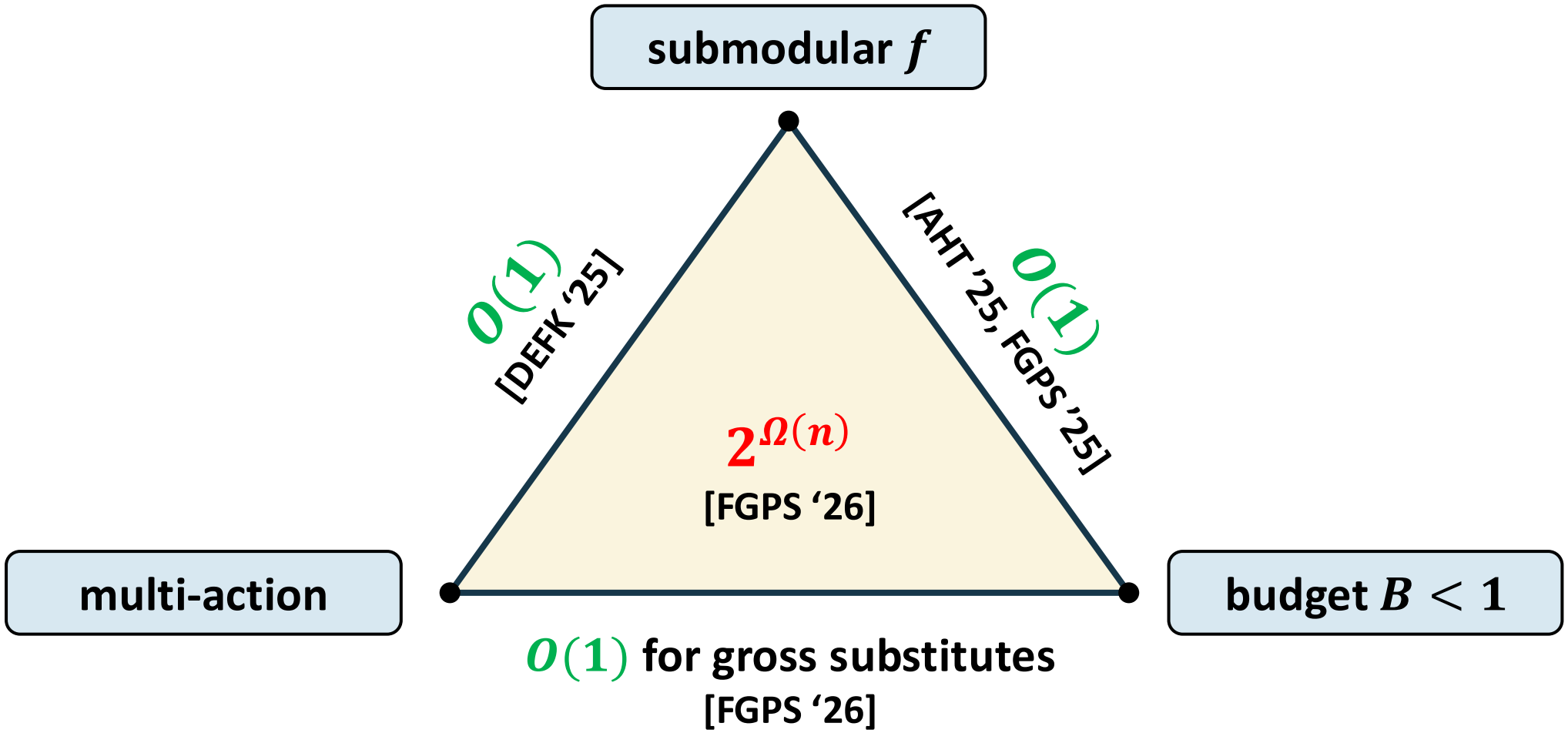
All negative results extend automatically.

Theorem [AHT '25, FPGS '25]: For binary actions,
all positive results extend to budgets.

Theorem [FPGS '26]: For multiple actions,
(a) separation: hardness result for submodular f with $B < 1$.
(b) $O(1)$-approximation for gross substitutes f with a budget.

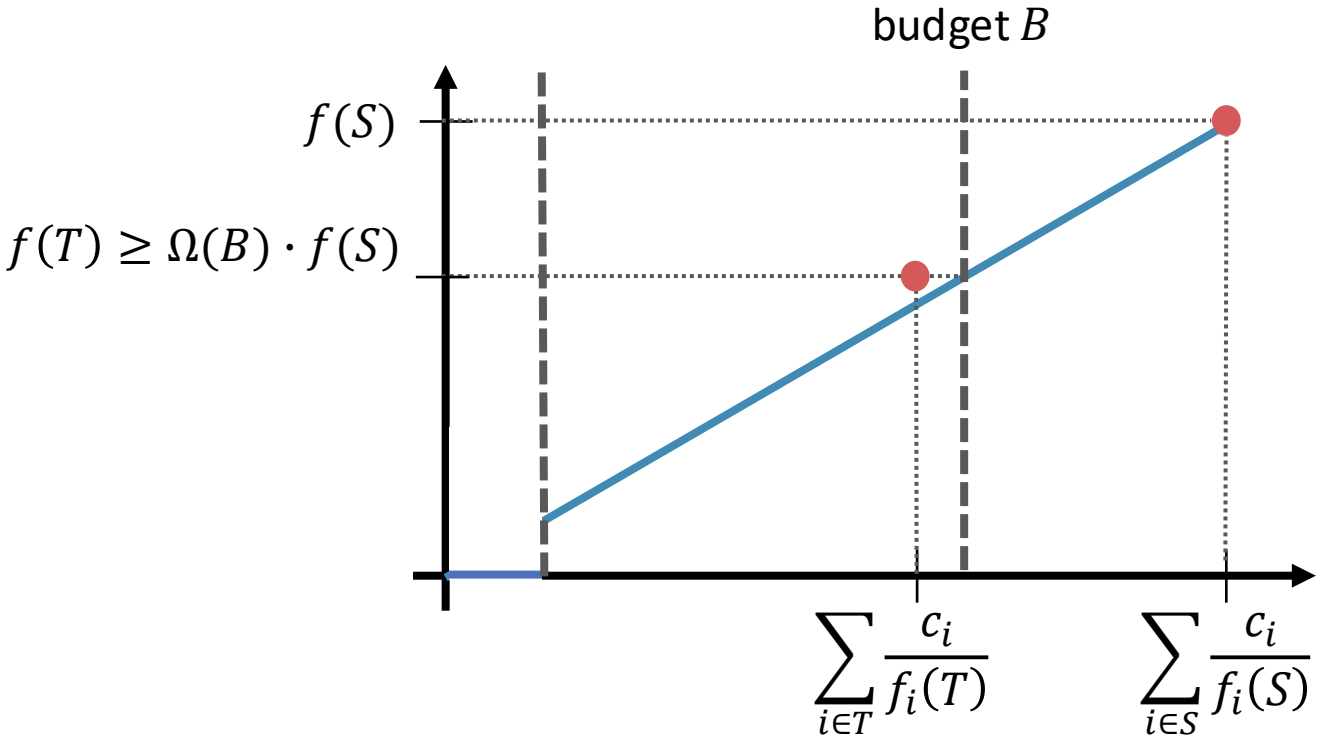
Budget-Feasible Contracts

Budget constraint: $\sum_{i\in[n]} \alpha_i \leq B$ for some $B \in [0,1]$.



Budgets with Binary Actions

Downsizing Lemma: For binary actions, XOS f, and any* budget B,
we can efficiently reduce any team S to a budget-feasible team $T \subseteq S$,
while incurring only a loss to reward that is linear in B.

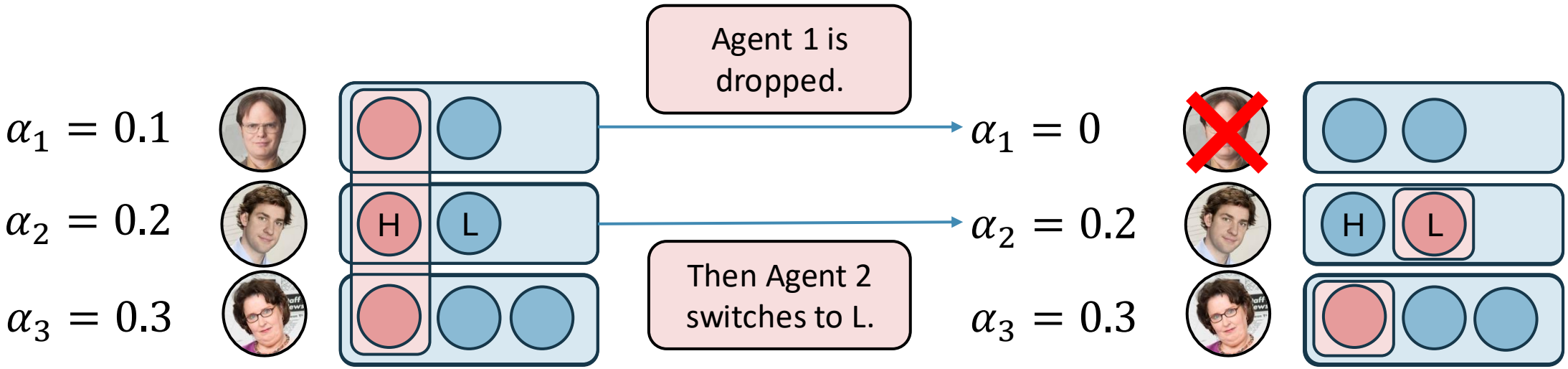


*up to the cost of incentivizing a single agent

Budgets with Multiple Actions

Best-Response Monotonicity: When an agent is dropped, each remaining agent will only add actions.

Recall that this property fails for multiple actions with submodular f.



Theorem: Best-response monotonicity holds for gross substitutes f.

This can be used to establish a downsizing lemma for multi-action.

Alternative Objectives

[AHT '25] [FGPS '25]

Principal's utility: $(1 - \sum_{i=1}^{n} \alpha_i) \cdot f(T)$

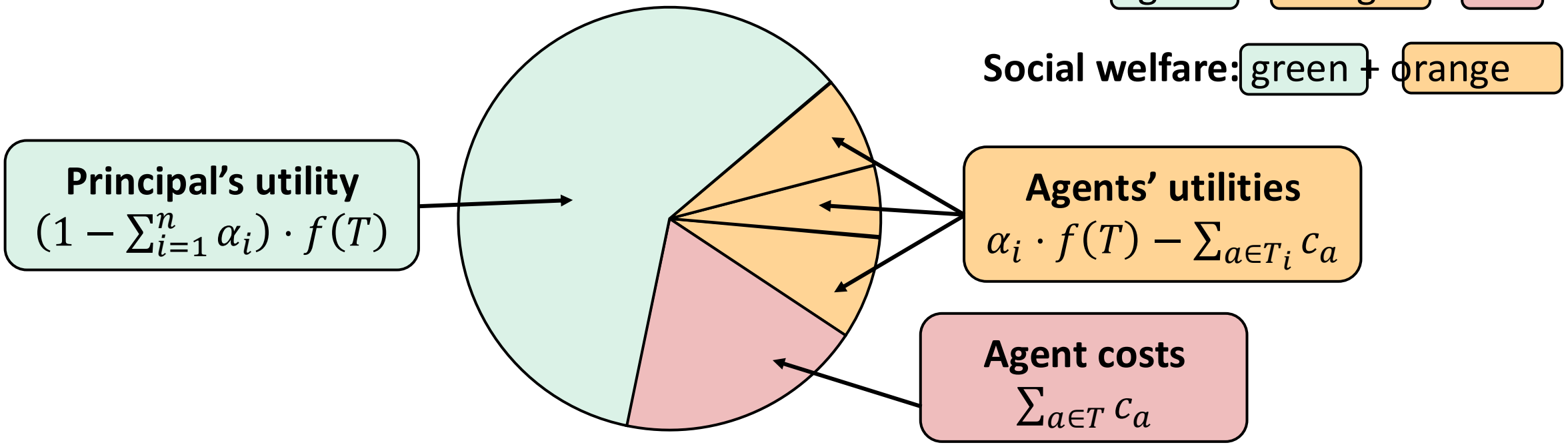
Social welfare: $f(T) - \sum_{a \in T} c_a$

Reward: $f(T)$

BEST objectives

Reward: green + orange + red

Social welfare: green + orange



Alternative Objectives

Principal's utility: $(1 - \sum_{i=1}^{n} \alpha_i) \cdot f(T)$

Social welfare: $f(T) - \sum_{a \in T} c_a$

Reward: $f(T)$

BEST objectives

Theorem [AHT '25, FPGS '25]: For binary actions, all $O(1)$-approximations extend to all BEST objectives.

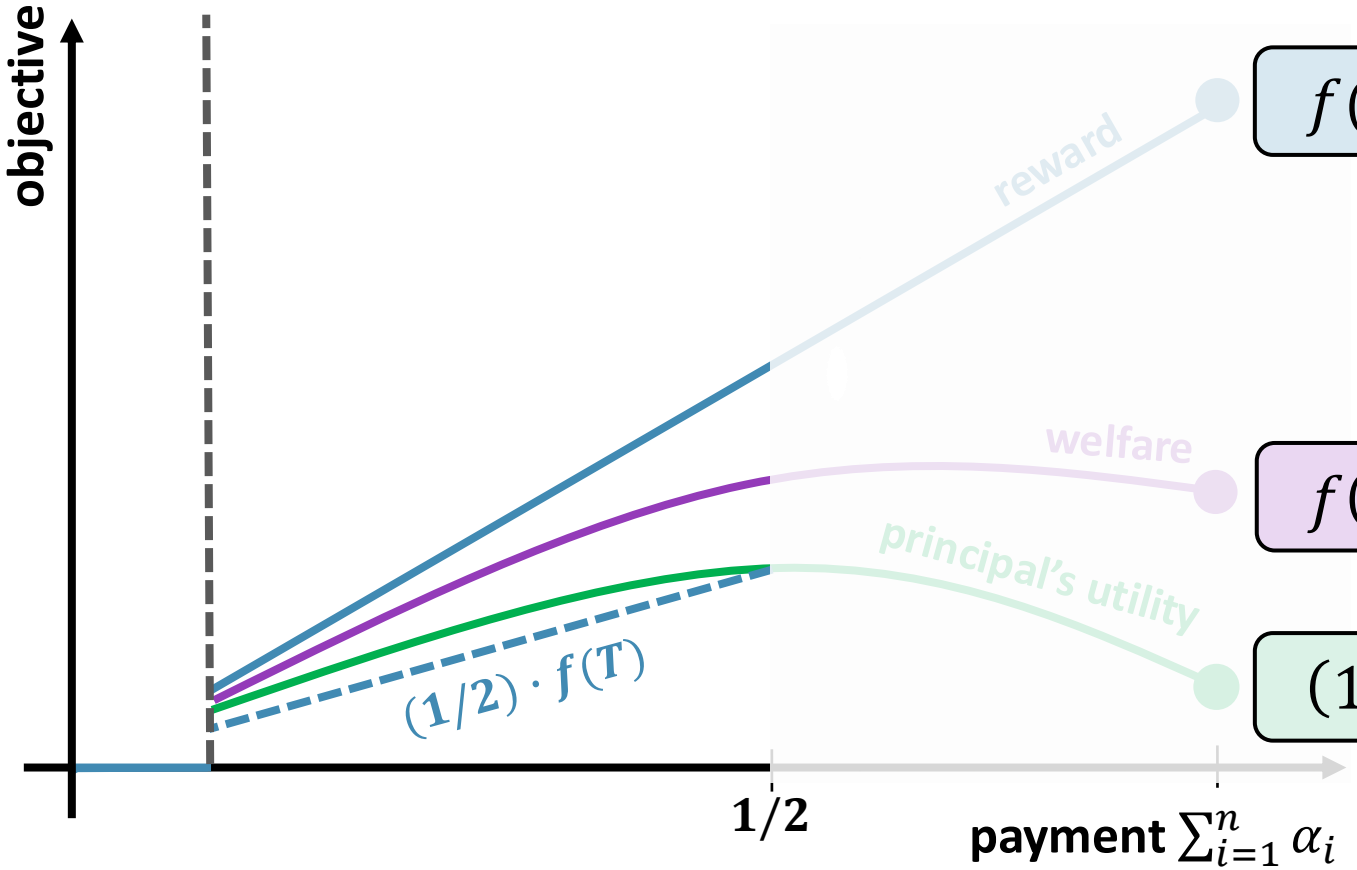
Theorem [FPGS '26]: For multiple actions, $O(1)$-approximation for gross substitutes f extends to all BEST objectives.

Multiple actions with submodular f:

Principal's utility	Social welfare	Reward
$O(1)$	?	No $2^{\Omega(n)}$

Alternative Objectives

Proof idea: For total payment $\sum_{i=1}^{n} \alpha_i \leq 1/2$,
reward, welfare, and principal's utility differ by factor ≤ 2



$f(T)$

$f(T) - \sum_{a \in T} c_a$

$(1 - \sum_{i=1}^{n} \alpha_i) \cdot f(T)$

Downsizing lemma shows that scaling the payment to $\leq 1/2$ loses at most an $O(1)$-factor*

*Handling expensive agents separately

Supermodular Rewards

Submodular:
$f_i(S) \leq f_i(T)$ for $S \supseteq T$

Submodular: The larger the team,
the smaller an individual contribution.

Supermodular:
$f_i(S) \geq f_i(T)$ for $S \supseteq T$

Supermodular: The larger the team,
the larger an individual contribution.

Supermodular Rewards

Theorem: For binary actions with supermodular f, obtaining any finite approximation is NP-hard via reduction from k-clique decision problem.

[Deo-Campo Vuong, Dughmi, Patel, Prasad, SODA 2024]

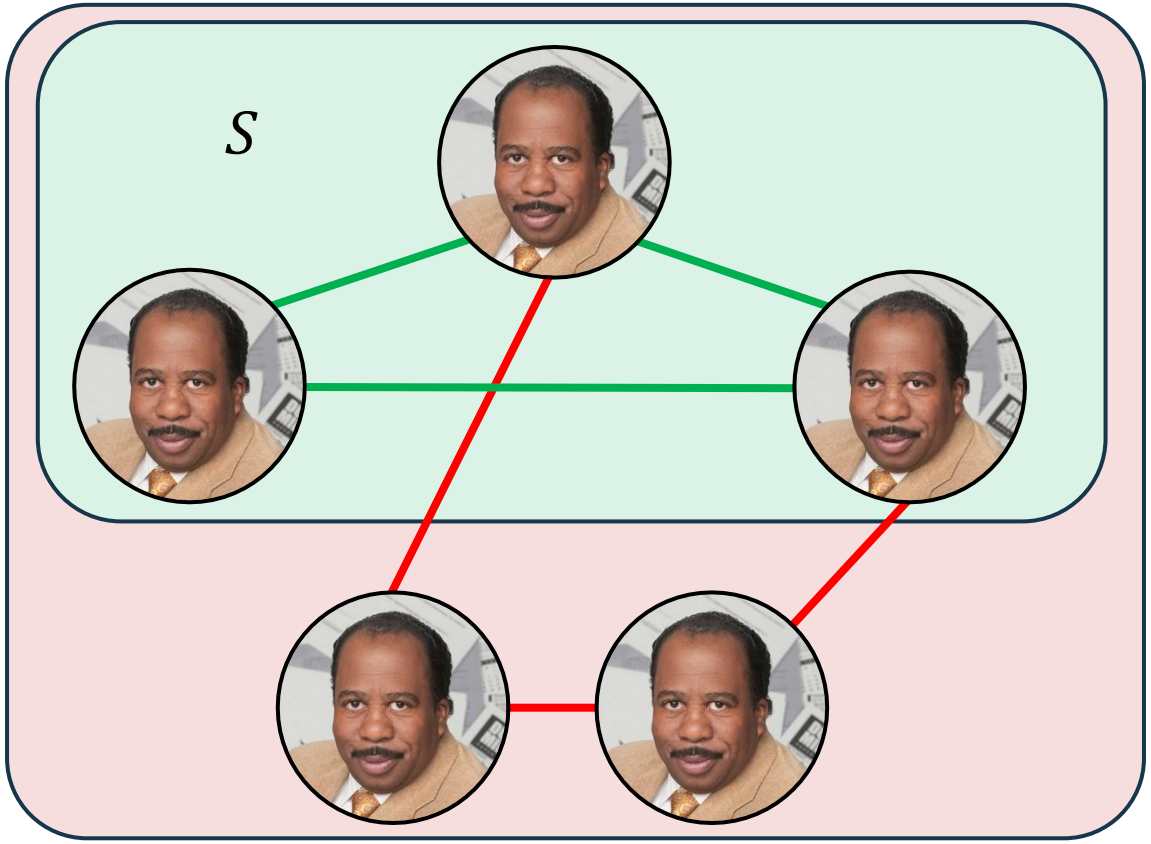
We set $f(S) = |\{\text{edge } (i,j) \mid i,j \in S\}|$ and $c_i = (k-2)/(k-1)$ for all i.

If S is a k-clique, then

$$g(S) = \left(1 - \sum_{i \in S} \frac{c_i}{f_i(S)}\right) \cdot f(S) > 0$$

because $\sum_{i \in S} \frac{c_i}{f_i(S)} = \frac{k \cdot (k-2)}{(k-1) \cdot (k-1)} < 1.$

If there is no k-clique, $g(S) \leq 0$ for all S.



Multiple Projects

Delegating multiple binary-outcome projects simultaneously.
Each agent can work on at most one project.

[Alon, Castiglioni, Chen, Ezra, Li, Talgam-Cohen, EC 2025]

All negative results extend automatically.

Theorem [ACC+ '25]: $O(1)$-approximation results for binary actions with submodular and XOS f extend to multiple projects.

Multiple Projects

Proof idea: A fractional LP relaxation of the problem.
Variable $y_{j,x,S} \in [0,1]$ represents the allocation of team S to project j,
and x is an estimate of the reward generated by that project.

Resembles LP relaxations for welfare maximization.

Exponentially many variables but only polynomially many constraints: Dual LP
can be solved using separation oracles,
which require solving approximate demand queries for $\hat{f} = \min(f, x)$.

Proof also handles heavy agents and rounding fractional solutions.

Equal-Pay Contracts

A contract $\alpha = (\alpha_1, \ldots, \alpha_n)$ is **equal-pay** if $\alpha_i = \alpha_j$ for i, j with $\alpha_i, \alpha_j \neq 0$.


$\alpha_1 = 0.2$


$\alpha_2 = 0.2$


$\alpha_3 = 0.2$


$\alpha_4 = 0$

In many hardness results for unconstrained contracts,
the optimal contract is equal-pay.

[DEFK '23] [EFS '24] [DEFK '25] [DDPP '24]

Theorem [FGPS '26, CCL '25]: $O(1)$-approximations for
multi-action and binary-action extend to equal-pay contracts.

Theorem [FGPS '26, DLS '26]: The price of equality is $\Theta(\log n / \log \log n)$.

The End

Open Questions?

Learning near-optimal multi-agent contracts without full information

Complexity in terms of other resources than computation (e.g. communication)

Agents approximately best respond (cf. [DRT ‘2019] [BGCMG ‘2025])

Are the near-optimal simple PNEs?

Learning dynamics of contracts: Do agents converge to an equilibrium?

Summary

[Deo-Campo Vuong, Dughmi, Patel, Prasad, SODA 2024]
[Ezra, Feldman, Schlesinger, ITCS 2024]
[Alon, Castiglioni, Chen, Ezra, Li, Talgam-Cohen, EC 2025]
[Aharoni, Hoefer, Talgam-Cohen, EC 2025]
[Castiglioni, Chen, Li, 2025 & 2025]
[Feldman, Gal-Tzur, Ponitka, Schlesinger, EC 2025 & ITCS 2026 & EC 2026]
[Doron-Arad, Shachnai, Shmerler, Talgam-Cohen, EC 2026]
[Dütting, Ezra, Feldman, Kesselheim, EC 2026]
[Ding, Li, Sun, 2026]
[Lavi, Shachnai, Talgam-Cohen, 2026]

Your work here?