Combinatorial Contracts

Michal Feldman

Tel-Aviv University

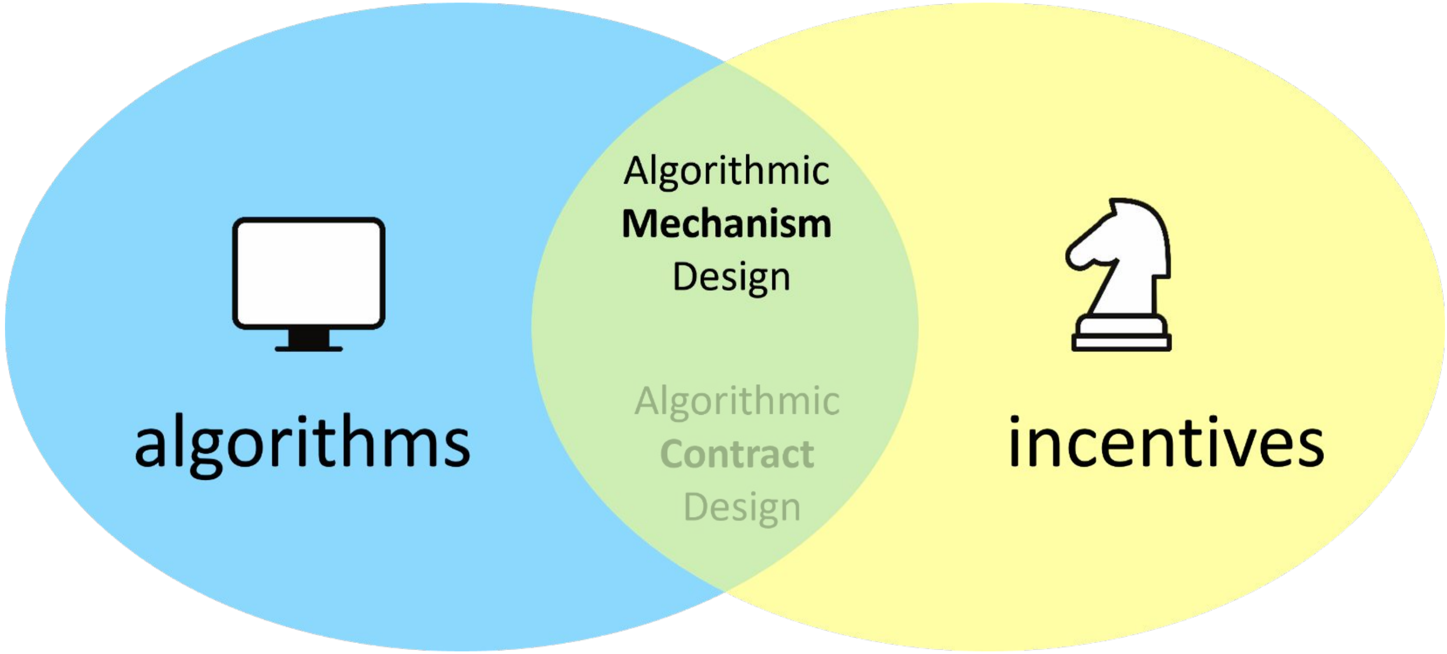
SLMath Workshop

November 8, 2023

Joint Work with:
Paul Duetting, Tomer Ezra, Yoav Gal Tzur, and Thomas Kesselheim
(FOCS’21, STOC’23, SODA’24)

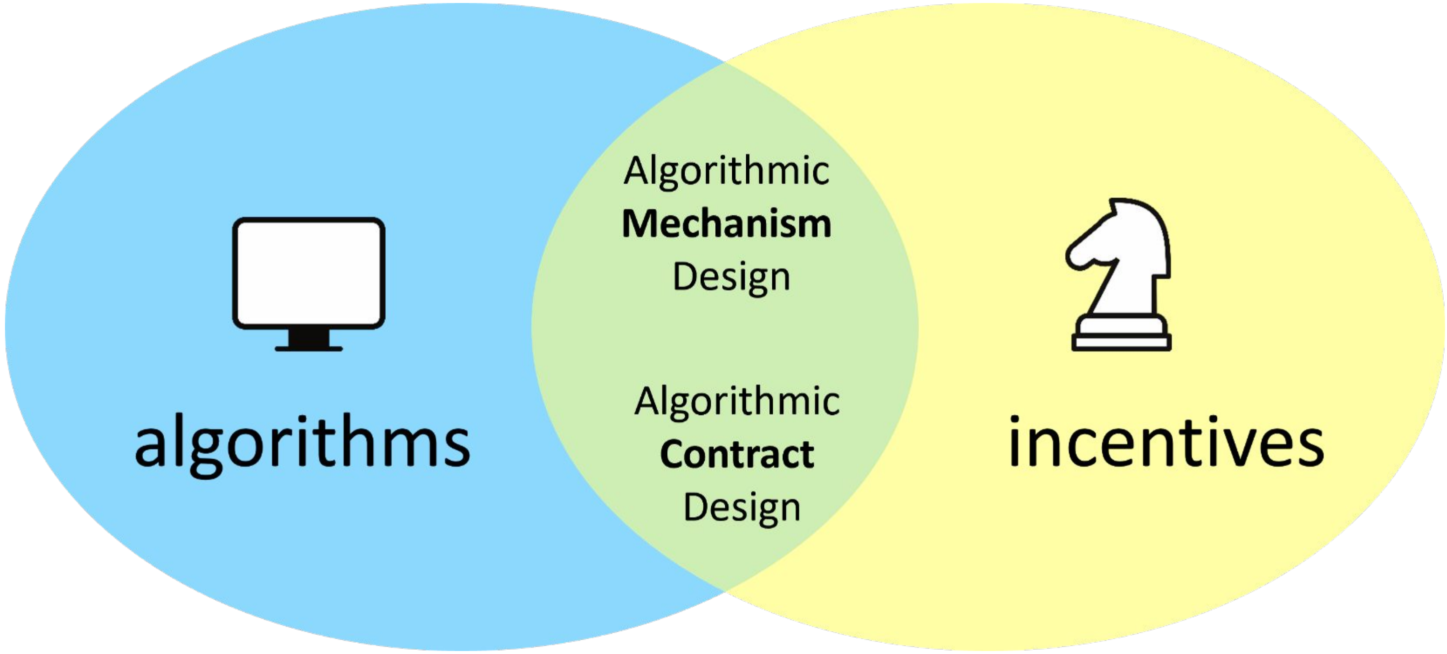
Algorithms and Incentives

Hidden preferences



Algorithms and Incentives

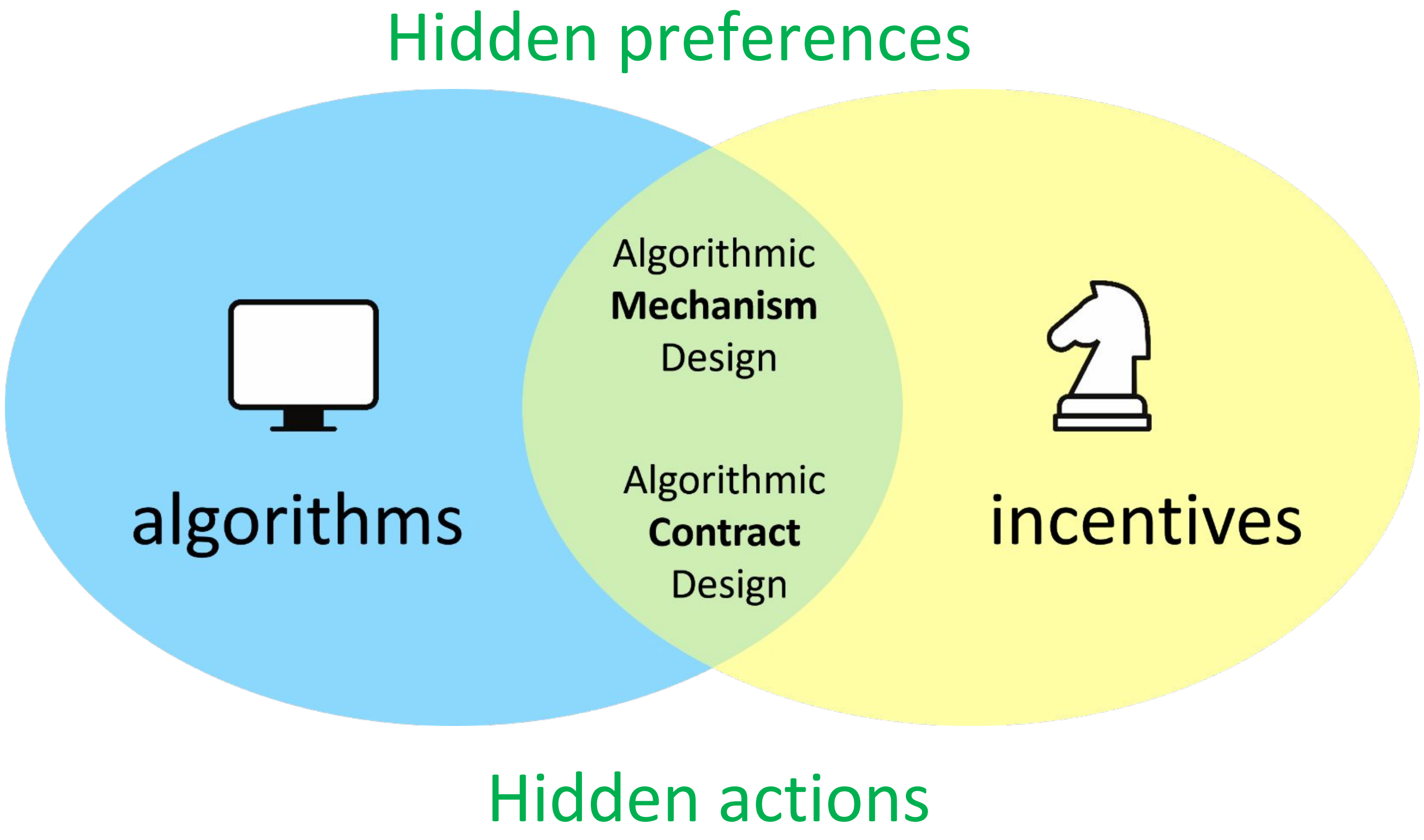
Hidden preferences



algorithms
Algorithmic **Mechanism** Design
Algorithmic **Contract** Design
incentives

Hidden actions

Algorithms and Incentives



Hidden preferences

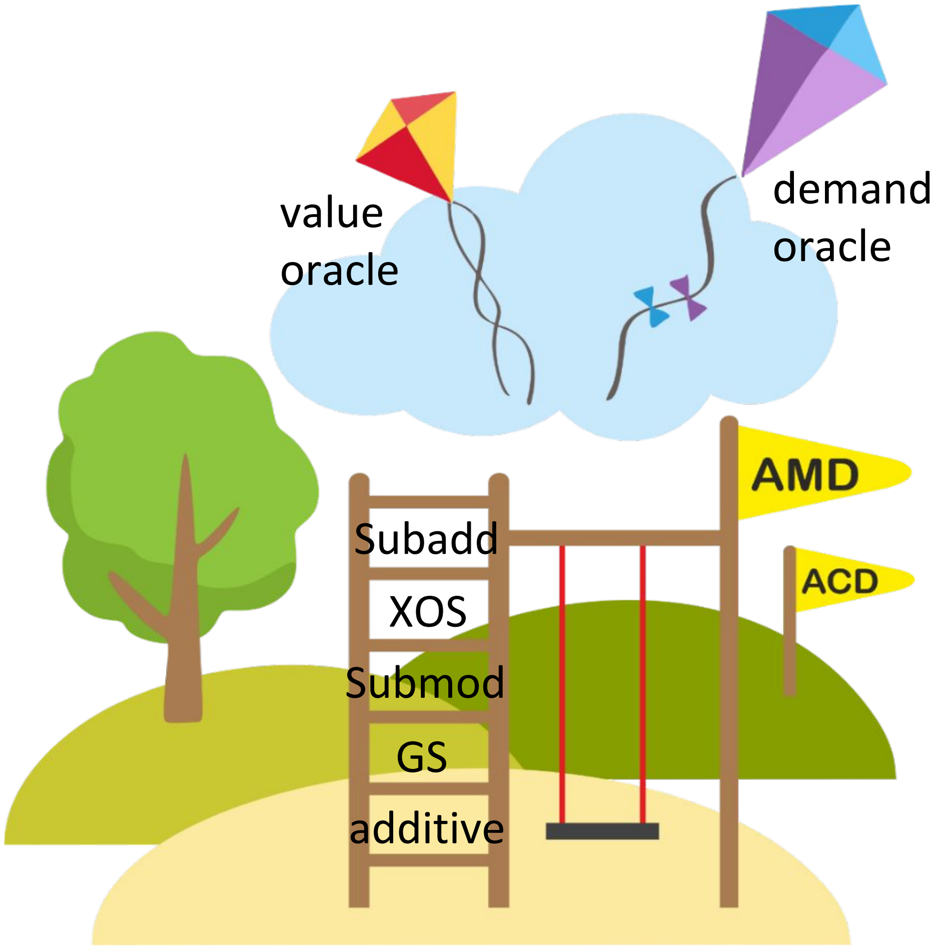
algorithms

Algorithmic **Mechanism** Design

Algorithmic **Contract** Design

incentives

Hidden actions



[Nisan Ronen 99]
[Lehmann Lehmann Nisan 01]
…

Within a Broader Perspective

An emerging frontier in Algorithmic Game Theory on optimizing the effort of others
(two recent workshops in STOC’22 and EC’22)

Contracts with multiple agents / multiple actions:

[Feldman Chuang Stoica Shenker EC’05] [Babaioff Feldman Nisan EC’06] [Emek Feldman WINE’09] [Babaioff Feldman Nisan Winter JET’12] [Dütting Ezra Feldman Kesselheim FOCS’21, STOC’23]

Contracts with multiple outcomes:

[Dütting Roughgarden Talgam Cohen EC’19] [Dütting Roughgarden & Talgam Cohen SODA’20] [Alon Dobson Procaccia Talgam Cohen Tucker-Foltz AAAI’20] [Alon Lavi Shamash Talgam Cohen EC’21] [Alon Dütting Talgam Cohen EC’21]

Optimal scoring rules: [Chen and Yu ’21] [Li et al., ‘22]

Delegation:

[Azar Micali TE’18] [Kleinberg Kleinberg EC’18] [Bechtel & Dughmi ITCS’21] [Braun Hahn Hoefer & Schecker ‘22]

Strategic classification:

[Kleinberg & Raghavan EC’19] [Ghalme Nair Eilat Talgam Cohen Rosenfeld ICML’21] [Nair Ghalme Talgam Cohen Rosenfeld ’22]

Emerging Frontier

- Simple vs optimal contracts: [Dutting Roughgarden & Talgam-Cohen EC’19], [Alon Dutting Li Talgam-Cohen EC’23]
- Combinatorial contracts: [Lavi & Shamash EC’19], [Dutting Roughgarden & Talgam-Cohen SODA’20], [Dutting Ezra F. & Kesselheim FOCS’21], [Alon Lavi Shamash & Talgam-Cohen EC’21], [Dutting Ezra F. & Kesselheim STOC’23], [Babaioff F. Nisan EC’12], [Castiglioni et al. EC’23], [Dutting F. & Gal-Tzur, SODA’24], [Ezra F. Schlesinger’23]
- Contract design for social goods: [Li Immorlica & Lucier WINE’11], [Ashlagi Li & Lo Management Science’23+]
- Typed contracts: [Guruganesh Schneider & Wang EC’21], [Alon Dutting & Talgam-Cohen EC’21], [Castiglioni et al. EC ‘21], [Castiglioni et al. EC ‘22], [Guruganesh Schneider & Wang EC’23]
- Learning contracts: [Ho Slivkins & Vaughn EC’14], [Cohen Deligkas & Koren SAGT’22], [Zhu et al. EC’23], [Dutting Guruganesh Schneider & Wang ICML’23]

"Optimizing the Effort of Others"

Contracts with multiple agents:

[Feldman Chuang Stoica Shenker EC'05] [Babaioff Feldman Nisan EC'06] [Emek Feldman WINE'09] [Babaioff Feldman Nisan Winter JET'12]

Contracts with multiple outcomes:

[Dütting Roughgarden Talgam Cohen EC'19] [Dütting Roughgarden & Talgam Cohen SODA'20] [Alon Dobson Procaccia Talgam Cohen Tucker-Foltz AAAI'20] [Alon Lavi Shamash Talgam Cohen EC'21] [Alon Dütting Talgam Cohen EC'21]

Contracts with multiple actions:

[Dütting Ezra Feldman Kesselheim FOCS'21]

Delegation:

[Azar Micali TE'18] [Kleinberg Kleinberg EC'18] [Bechtel & Dughmi ITCS'21] [Braun Hahn Hoefer & Schecker '22]

Strategic classification:

[Kleinberg & Raghavan EC'19] [Ghalme Nair Eilat Talgam Cohen Rosenfeld ICML'21] [Nair Ghalme Talgam Cohen Rosenfeld '22]

Algorithmic Contract Design within a Broader Perspective

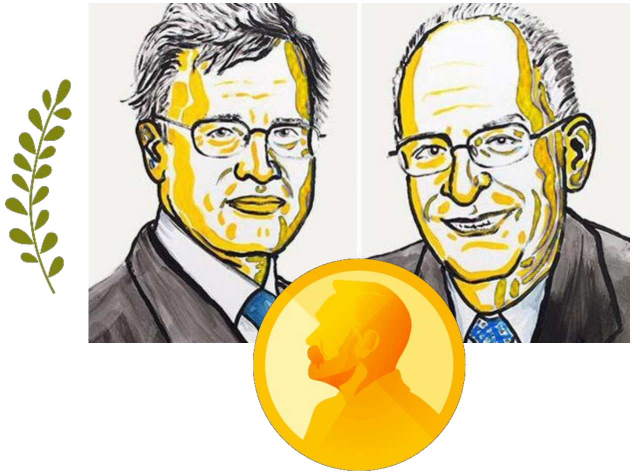
An emerging frontier in Algorithmic Game Theory on optimizing the effort of others (two recent workshops in STOC’22 and EC’22)

- Algorithmic contract design [e.g., Dutting et al., 2019, 2021a,b]
- Strategic classification [e.g., Kleinberg and Raghavan, 2019; Bechavod et al., 2022
- Optimal scoring rule design [e.g., Chen and Yu, 2021; Li et al., 2022]
- Delegation [e.g., Kleinberg and Kleinberg, 2018; Bechtel et al., 2022]
- …

Contract Design

One of the pillars of microeconomic theory
[Ross’73, Holmstrom’79]

Holmstrom Hart



“The 2016 Nobel Prize in Economics was awarded Monday to Oliver Hart and Bengt Holmström for their work in contract theory — **developing a framework to understand agreements like insurance contracts, employer-employee relationships and property rights**.”

- As markets for services move online, they grow in scale and complexity (freelance services, legal services, marketing services, etc.)
- An algorithmic / computational approach is timely and relevant

The Algorithmic/Computational Lens

- The algorithmic lens has been traditionally useful
  - Reveals structure
  - Identifies tractability frontier
  - Informs the design of simple mechanisms

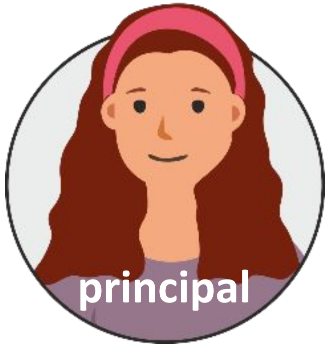
- Many examples in Algorithmic **Mechanism Design**
  - E.g., greedy algorithms, substitutes as a frontier of tractability, submodularity as simplicity frontier, hardness of NE, …

- Study **Contract Design** from a computational/algorithmic perspective

Principal-Agent Model

I won't be able to monitor his work. Who knows? he might go to the beach instead of focusing on the event

Organizing this event is gonna be so much work. I'll need to talk with speakers, manage the schedule, do logistics, …

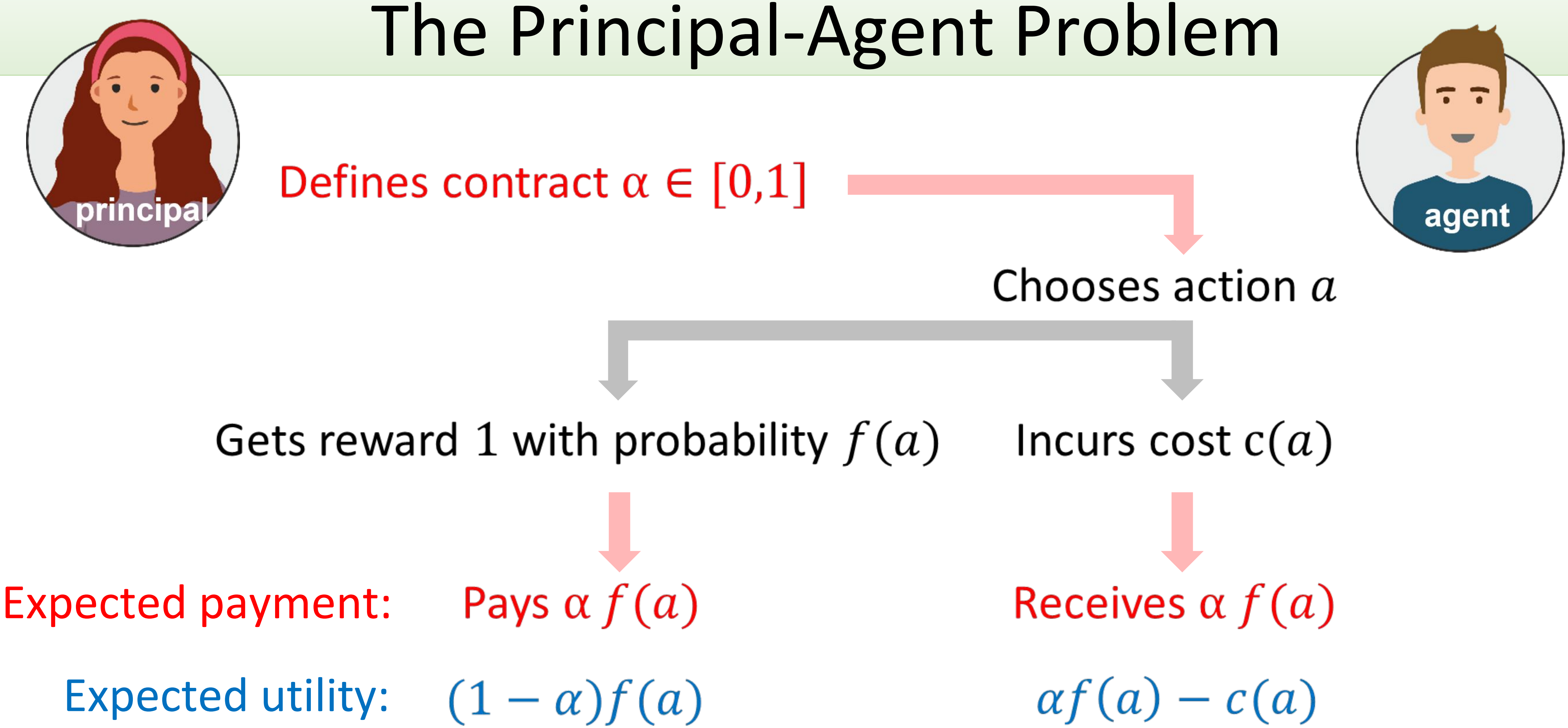


Would you please organize TCS+ for me?

I'll only pay you if the event turns out to be a huge success

How much would you pay me?

The Principal-Agent Problem



Defines contract α ∈ $[0,1]$

Chooses action a

Gets reward 1 with probability $f(a)$

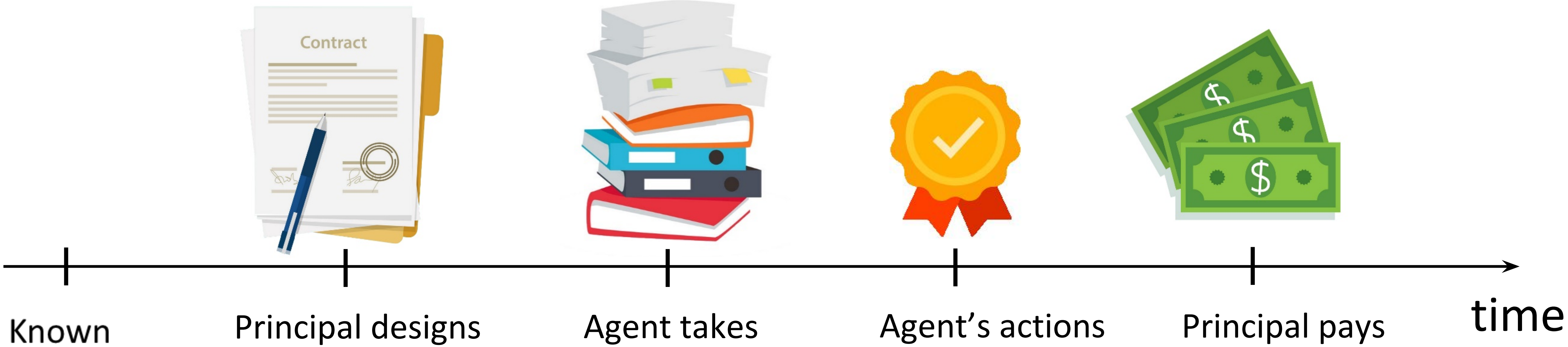
Incurs cost c(a)

Expected payment: Pays α $f(a)$ Receives α $f(a)$

Expected utility: $(1-\alpha)f(a)$ $\alpha f(a) - c(a)$

Defining features: hidden action, stochastic outcome, limited liability

The Principal-Agent Problem



Known setting: f, c

Principal designs a contract (payment for every outcome)

Agent takes unobserved costly actions

Agent’s actions produce an outcome, stochastically

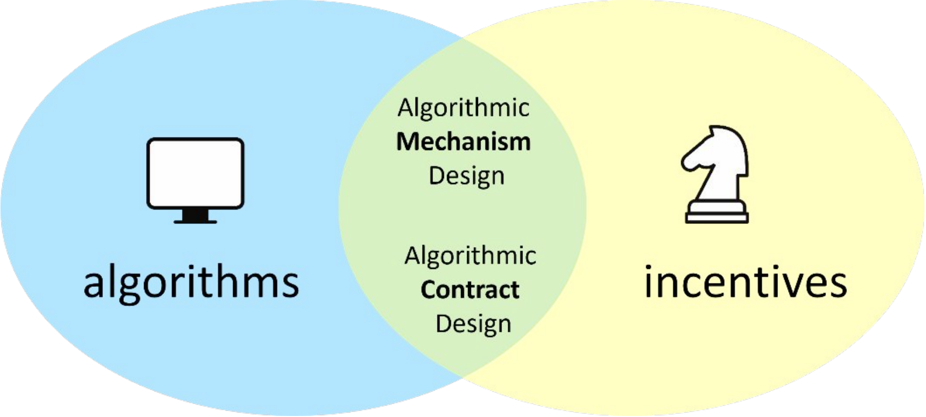
Principal pays the agent based on observed outcome

Objective: maximize the expected utility of the principal

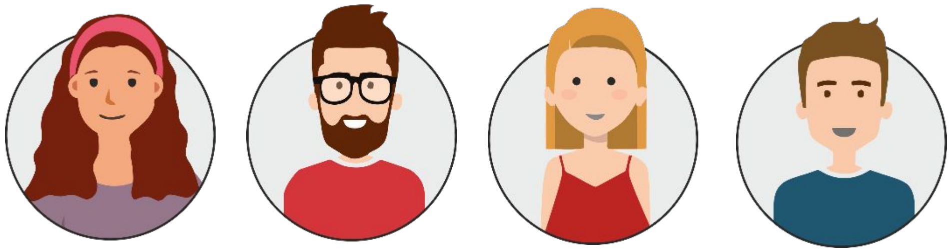
Many Additional Examples

- Freelance services
- Legal services
- Marketing services
- Course grading
- ..

As contracts move online, they're growing in scale and complexity

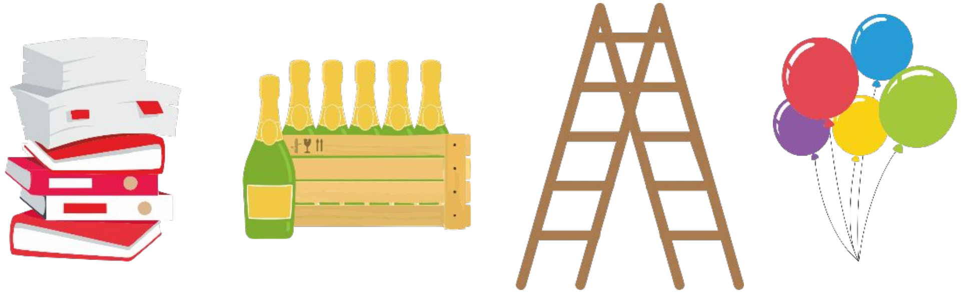

algorithms
Algorithmic **Mechanism** Design
Algorithmic **Contract** Design
incentives

Sources of Complexity in Contract Design



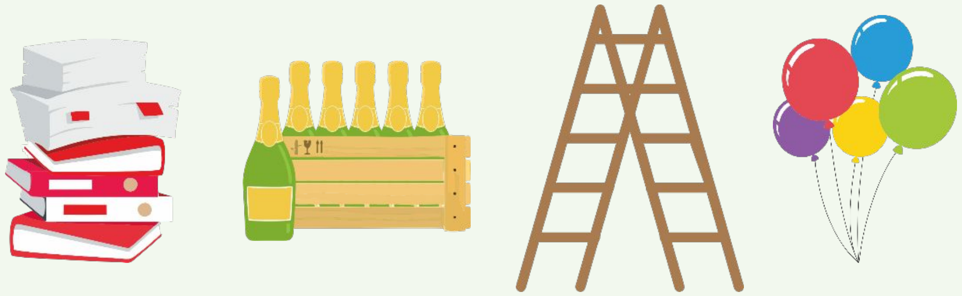
Multiple agents

[F Chuang Stoica Shenker EC’05,
Babaioff F Nisan EC’06, Emek F WINE‘09,
Ezra Duetting F Kesselheim, STOC’23]



Multiple actions

[Ezra Duetting F Kesselheim FOCS’21,
Duetting Feldman Gal-Tzur SODA’24,
Ezra F Schlesigner 2023]



Multiple actions

[Ezra Duetting F Kesselheim FOCS’21,
Duetting Feldman Gal-Tzur SODA’24,
Ezra F Schlesigner 2023]

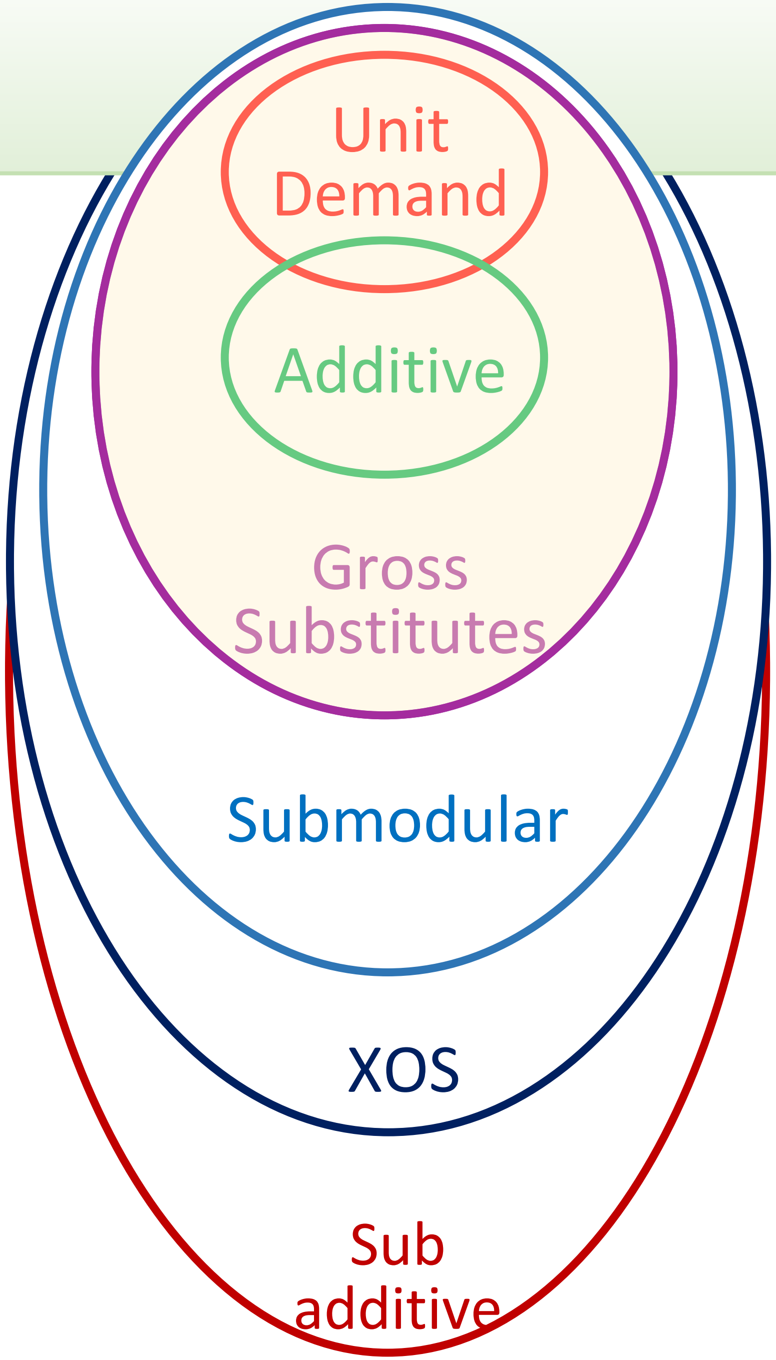
Single Agent, Many Actions

- n actions $A = \{1, \ldots, n\}$, agent chooses a set S

- $\mathrm{c(a)} \geq 0$: cost of action a
- $\mathrm{c}(S) = \sum_{a \in S} c(a)$ [additive cost]

- $f \colon 2^A \rightarrow [0,1]$ success probability function
  - $f(S)$: success probability for actions $S \subseteq A$
  - Not necessarily additive
- Reward: 1 for success, 0 for failure

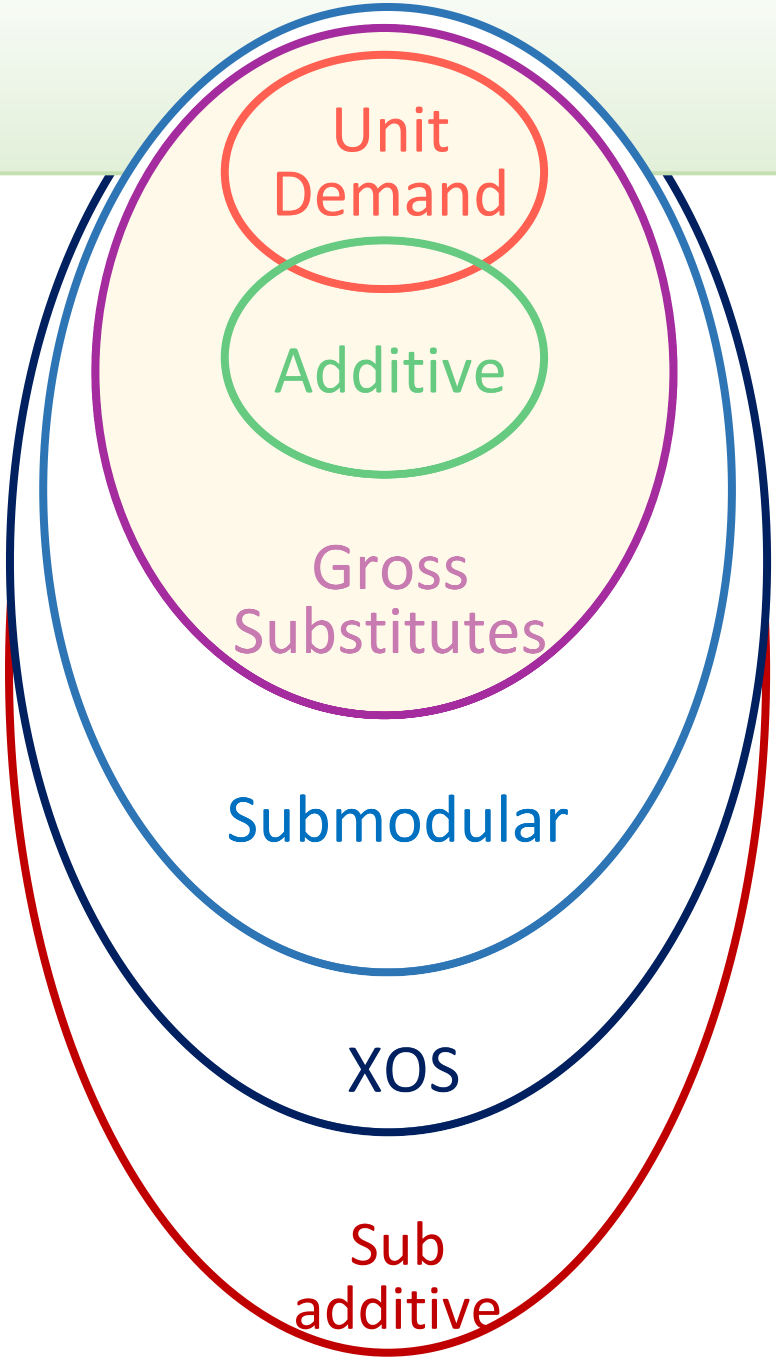
Submodular: $f(j \mid S) \geq f(j \mid T)$ for $S \subseteq T, j$ not in T
(decreasing marginal value)
Subadditive: $f(S) + f(T) \geq f(S \cup T)$



Single Agent, Many Actions

- n actions $A = \{1, \ldots, n\}$, agent chooses a set S
- $\mathrm{c(a)} \geq 0$: cost of action a
- $\mathrm{c}(S) = \sum_{a \in S} c(a)$ [additive cost]
- $f\colon 2^A \rightarrow [0,1]$ success probability function
  - $f(S)$: success probability for actions $S \subseteq A$
  - Not necessarily additive
- Reward: 1 for success, 0 for failure

Optimal Contract Problem (2-stage):
Find α that maximize $(1-\alpha)f(S)$ [principal's utility]
where S maximizes $\alpha f(S) - c(S)$ [agent's utility]



Computing Optimal Contracts

- If the agent only takes one action: Simple polynomial-time algorithm
  - For each action find smallest α that incentives the agent to take this action (e.g., by solving a poly-size LP)
  - Among these: Choose contract that maximizes the principal's expected utility

- **Our question:** What if the agent takes multiple actions?

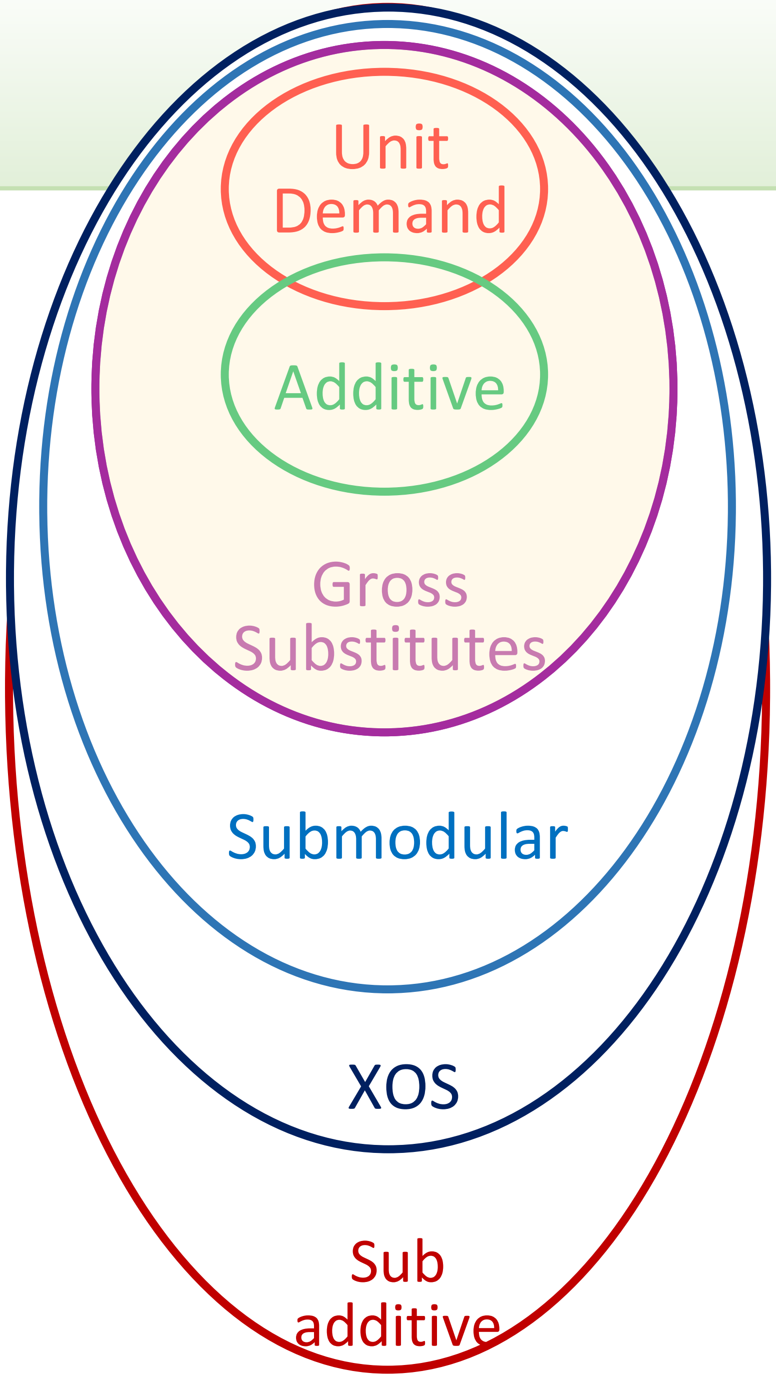
- Agent has 2^n choices $\Rightarrow$ Naïve approach inefficient

Main Results

Gross substitutes: for every price vectors $p \leq p'$ and a set $S \in D(f,p)$ there exists $S' \in \boldsymbol{D(f,p')}$ that includes all elements whose price did not increase

$\boldsymbol{D(f,p)}$: demand set ---
sets S maximizing utility $\mathrm{u}(S) = f(S) - \sum_{j \in S} p_j$

Useful property: for every price vector p, adding elements with maximal marginal utility greedily returns a set in $D(f,p)$



Main Results

Theorems

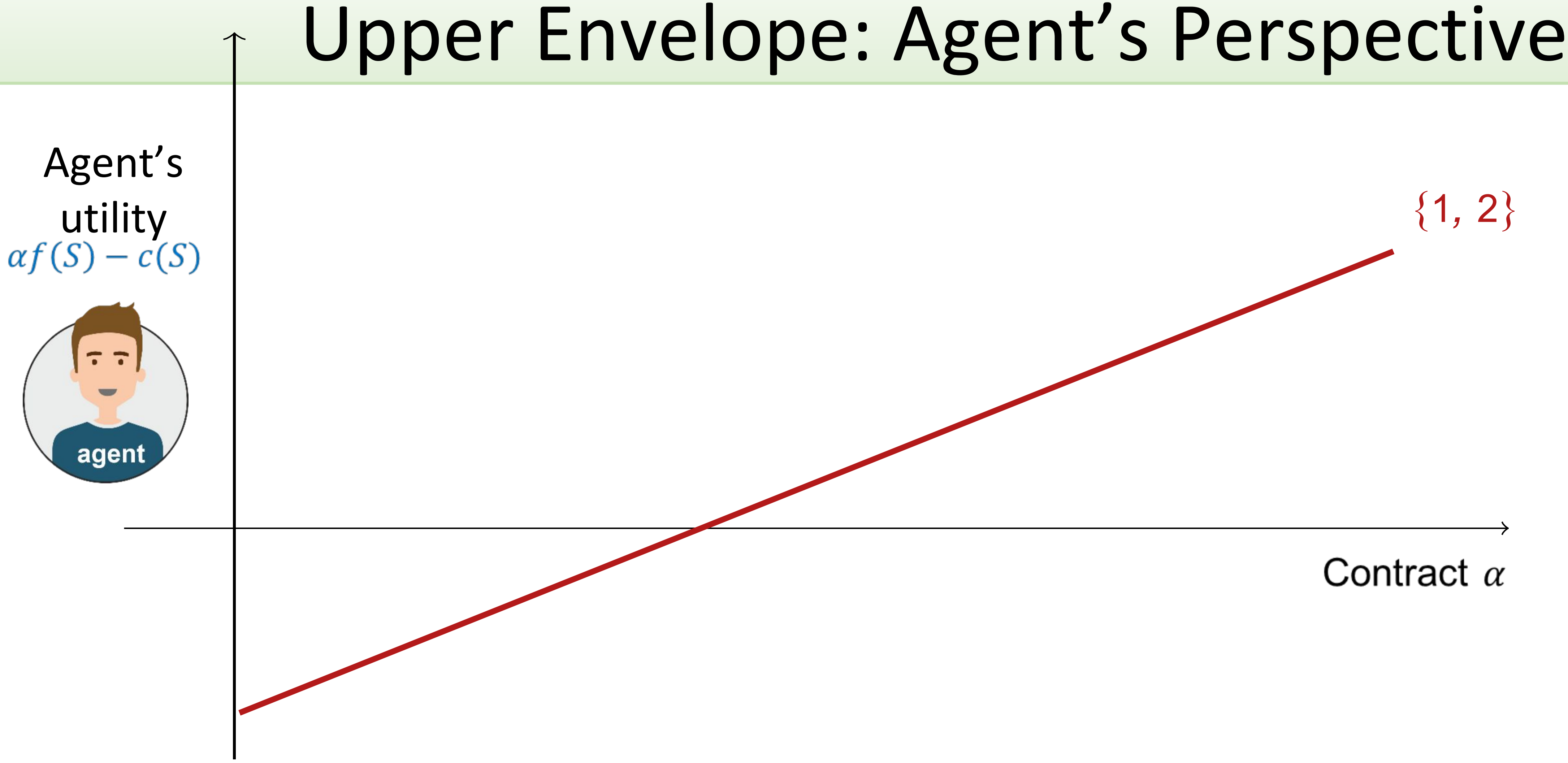
- A polynomial-time algorithm for gross substitutes functions
- For submodular functions (i.e., decreasing marginal contribution), it is NP-hard to compute the optimal contract

Gross substitutes constitutes a frontier, similar to

- welfare maximization tractability in combinatorial auctions [Nisan Segal 2006]

- market equilibrium existence [Kelso Crawford 1982, Gul Stacchetti 1999]

Upper Envelope: Agent’s Perspective

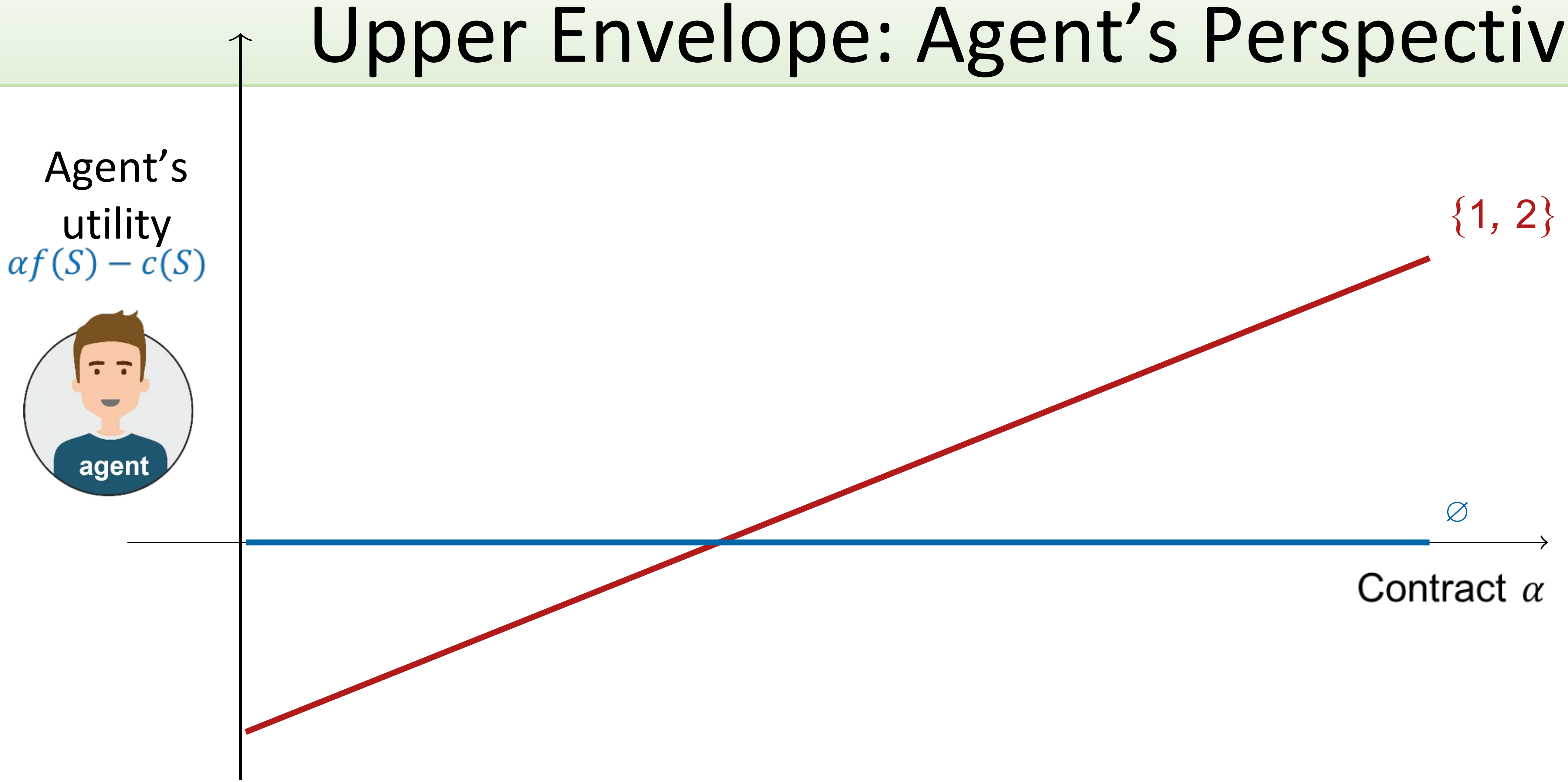


$$c(1) = 0.1, c(2) = 0.1, c(3) = 0.4, c(S) = \sum_{a \in S} c(a)$$

$$f(1) = 0.3, f(2) = 0.2, f(3) = 0.5, f(S) = \sum_{a \in S} f(a)$$

(additive)

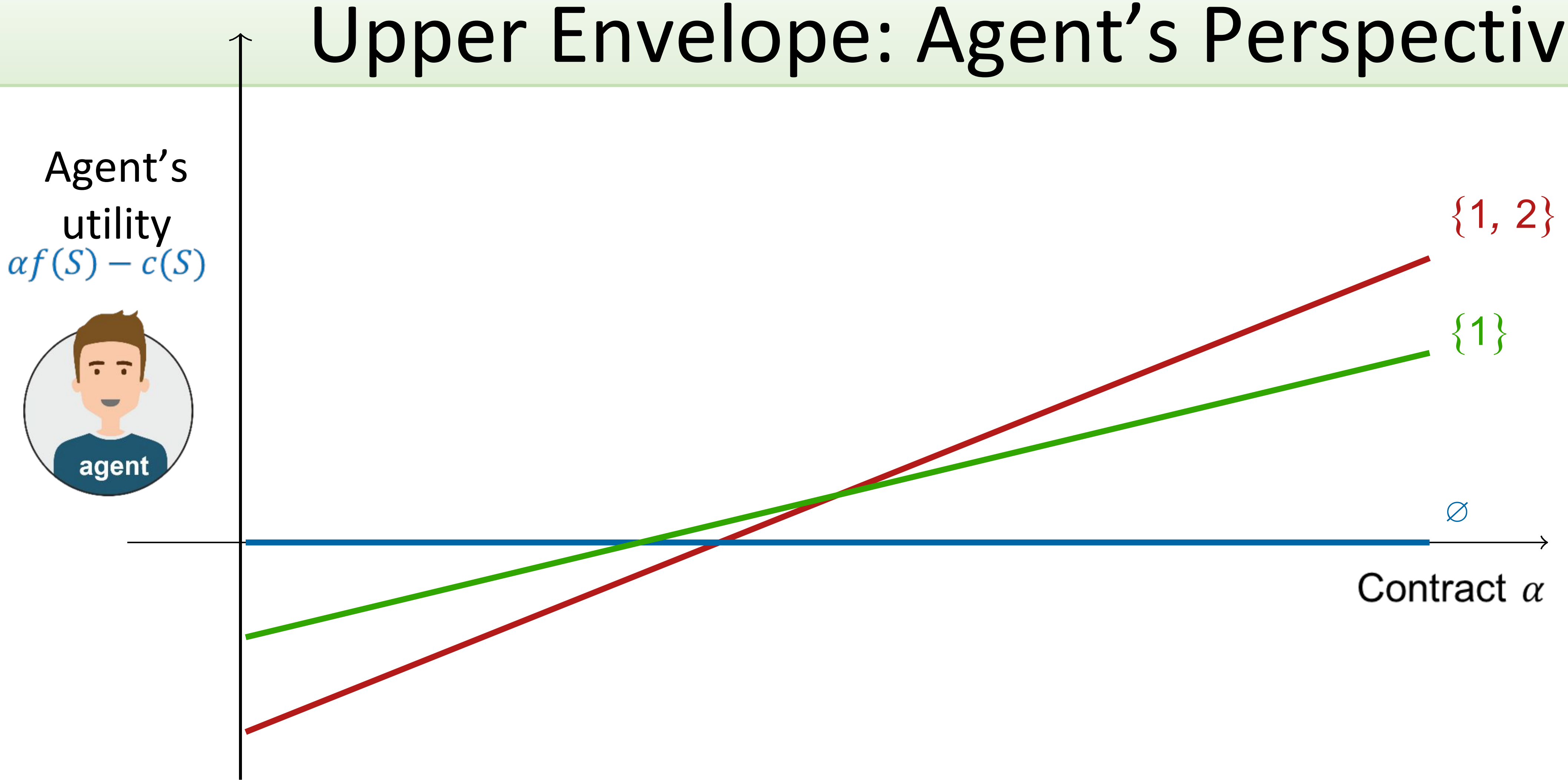
Upper Envelope: Agent’s Perspective



$c(1) = 0.1, c(2) = 0.1, c(3) = 0.4, c(S) = \sum_{a \in S} c(a)$

$f(1) = 0.3, f(2) = 0.2, f(3) = 0.5, f(S) = \sum_{a \in S} f(a)$ (additive)

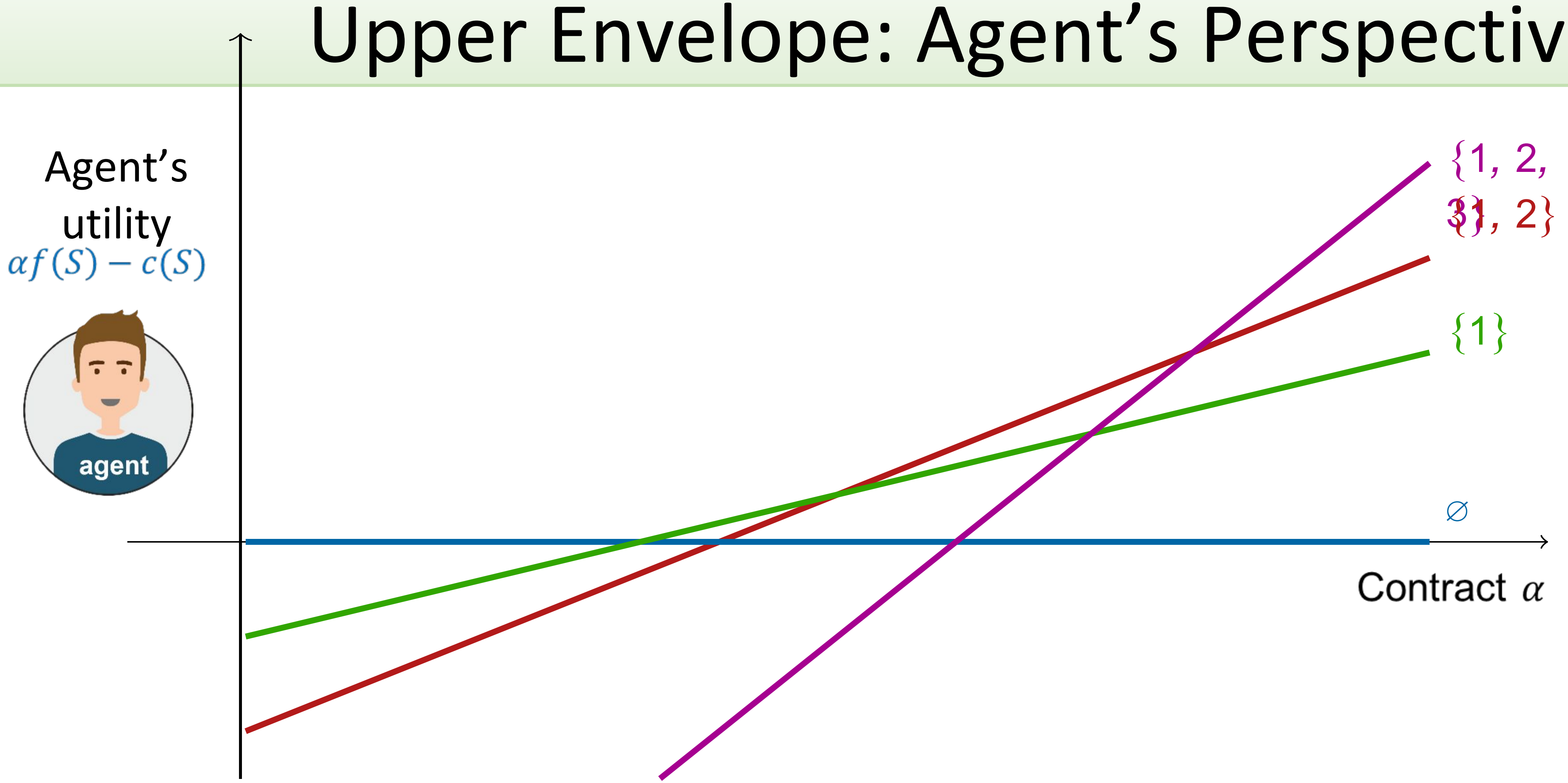
Upper Envelope: Agent's Perspective



$$c(1) = 0.1, c(2) = 0.1, c(3) = 0.4, c(S) = \sum_{a \in S} c(a)$$

$$f(1) = 0.3, f(2) = 0.2, f(3) = 0.5, f(S) = \sum_{a \in S} f(a)$$ (additive)

Upper Envelope: Agent’s Perspective

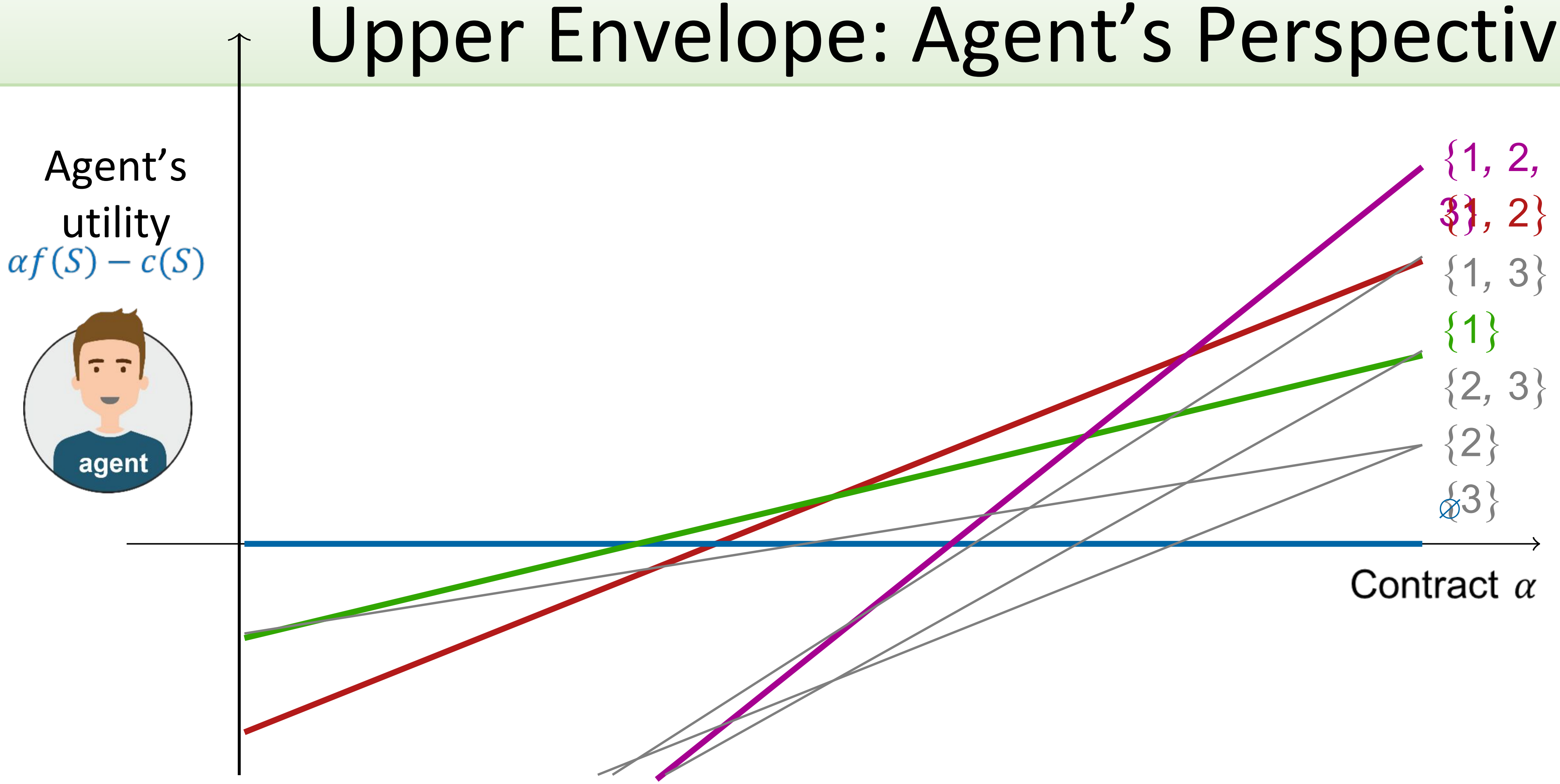


$$c(1) = 0.1, c(2) = 0.1, c(3) = 0.4, c(S) = \sum_{a \in S} c(a)$$

$$f(1) = 0.3, f(2) = 0.2, f(3) = 0.5, f(S) = \sum_{a \in S} f(a)$$

(additive)

Upper Envelope: Agent’s Perspective

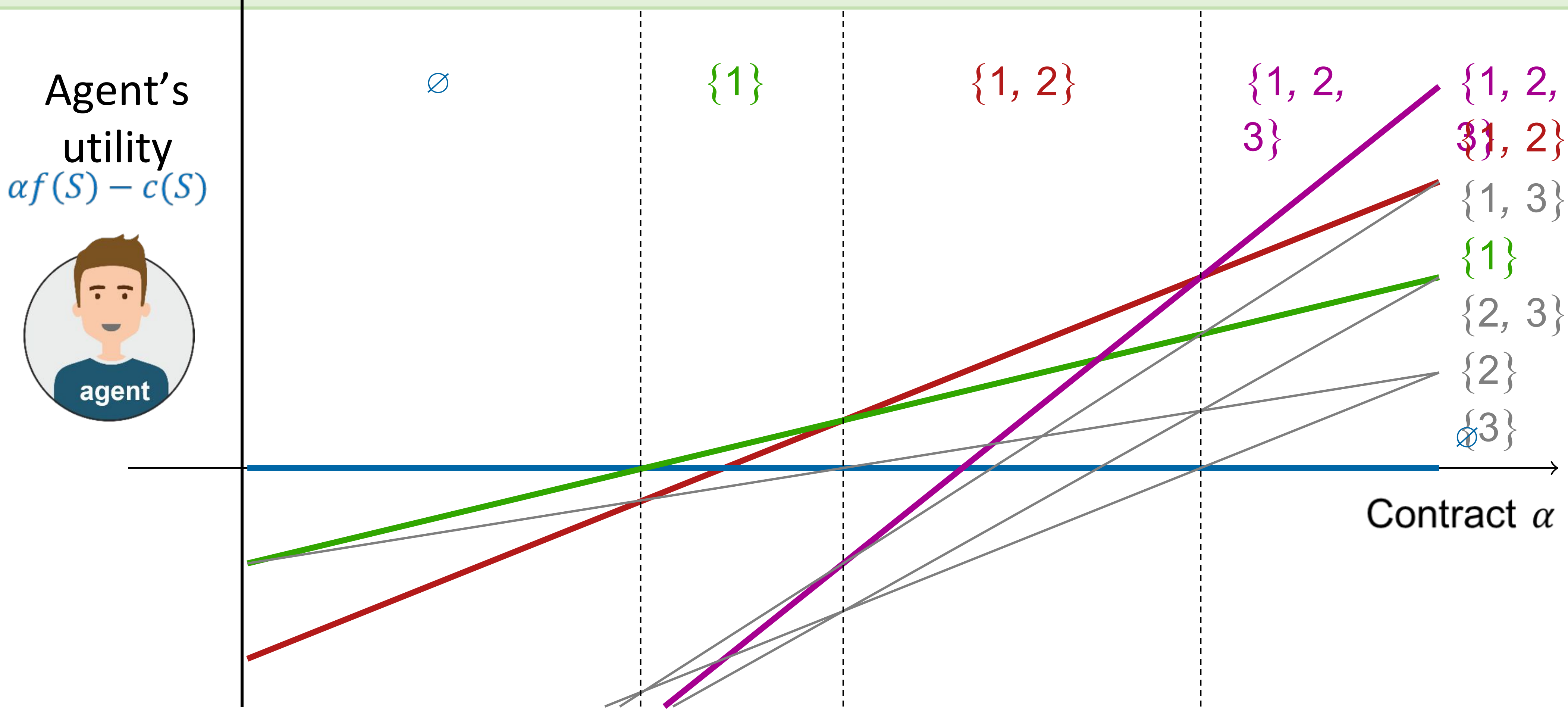


$$c(1) = 0.1, c(2) = 0.1, c(3) = 0.4, c(S) = \sum_{a \in S} c(a)$$

$$f(1) = 0.3, f(2) = 0.2, f(3) = 0.5, f(S) = \sum_{a \in S} f(a)$$

(additive)

Upper Envelope: Agent’s Perspective

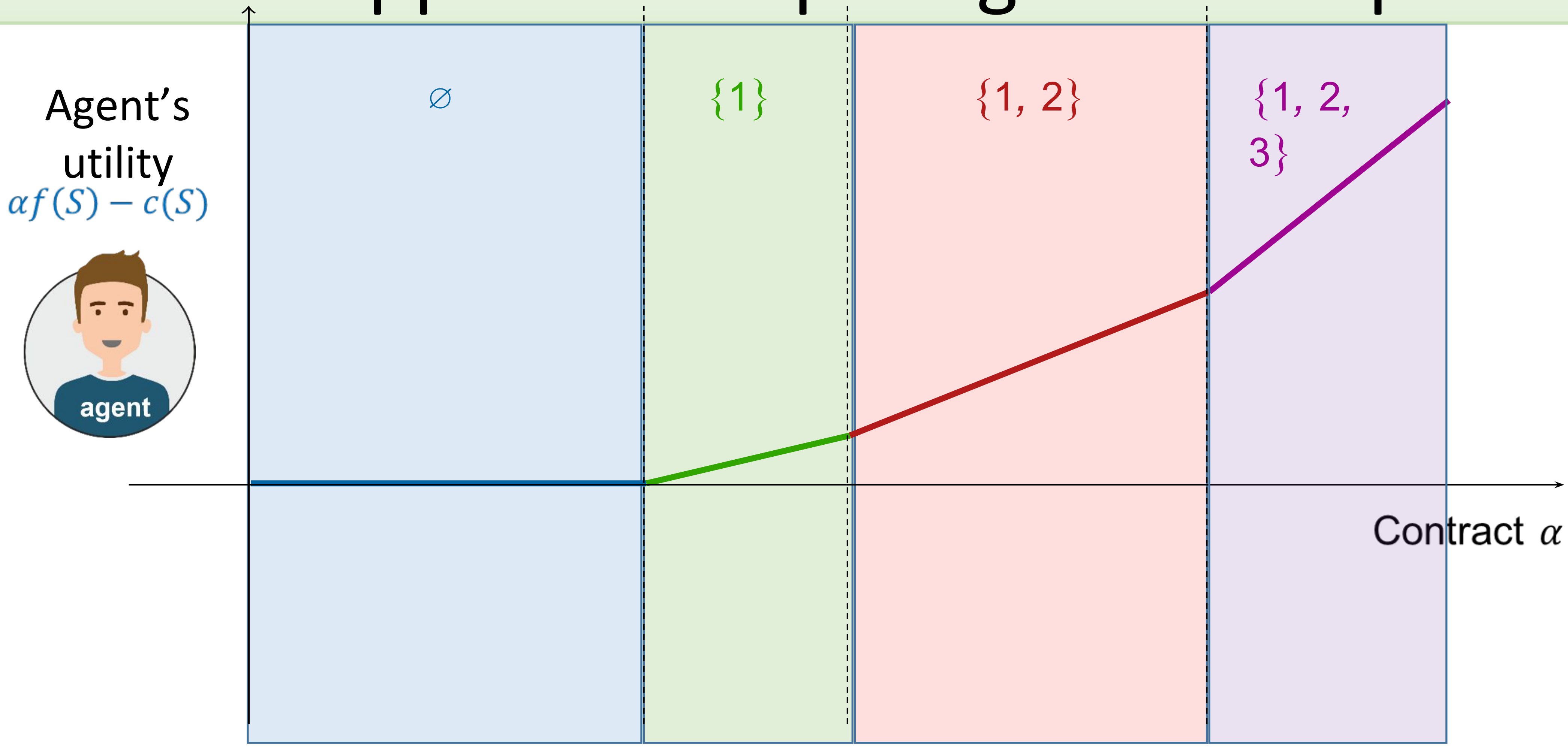


$$c(1) = 0.1, c(2) = 0.1, c(3) = 0.4, c(S) = \sum_{a \in S} c(a)$$

$$f(1) = 0.3, f(2) = 0.2, f(3) = 0.5, f(S) = \sum_{a \in S} f(a)$$

(additive)

Upper Envelope: Agent’s Perspective

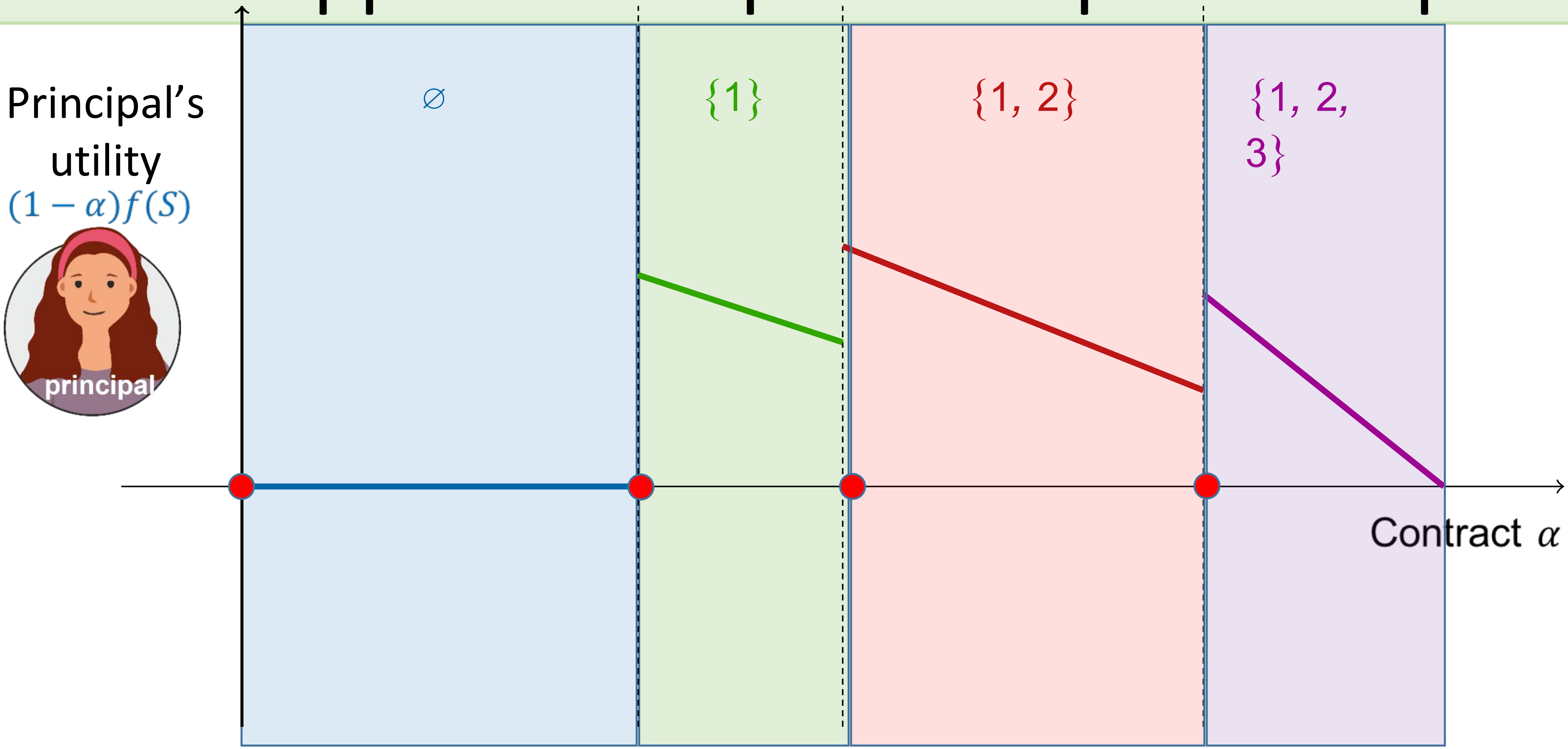


$$c(1) = 0.1, c(2) = 0.1, c(3) = 0.4, c(S) = \sum_{a \in S} c(a)$$

$$f(1) = 0.3, f(2) = 0.2, f(3) = 0.5, f(S) = \sum_{a \in S} f(a)$$

(additive)

Upper Envelope: Principal's Perspective



$c(1) = 0.1, c(2) = 0.1, c(3) = 0.4, c(S) = \sum_{a \in S} c(a)$

$f(1) = 0.3, f(2) = 0.2, f(3) = 0.5, f(S) = \sum_{a \in S} f(a)$ (additive)

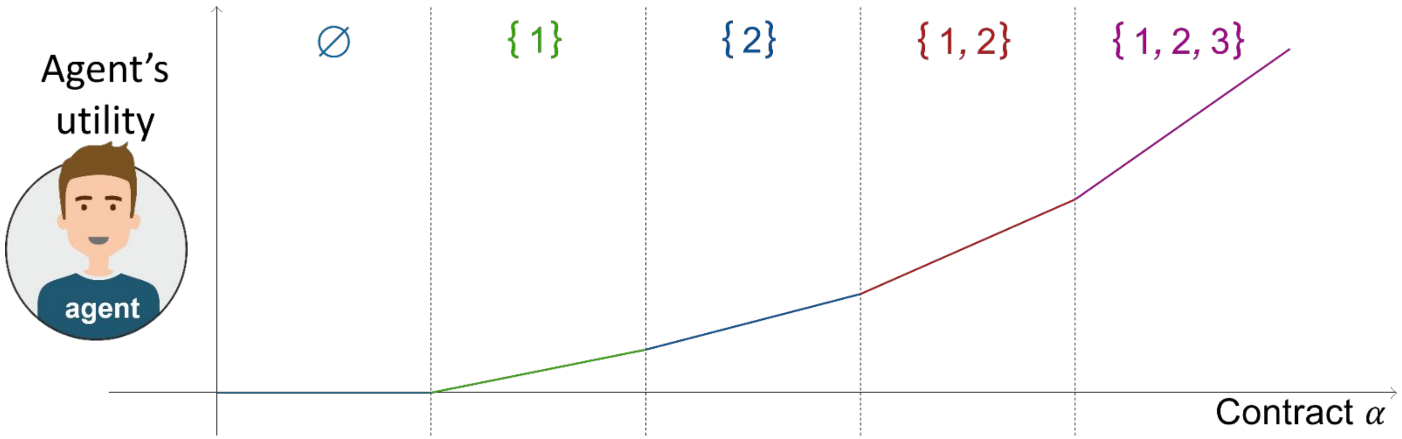
“Critical α’s” and an Algorithm

- **Simple observation**: can restrict attention to set of critical α’s

  (i.e., transition points of agent’s best response)
- Naïve algorithm: Go over all critical alphas and take the best; requires:
  - computing next critical alpha
  - computing agent’s best response
  - an upper bound on number of critical alphas

Theorem: For gross substitutes f, this yields a polynomial-time algorithm

Computing Optimal Contract for GS Functions



The agent's problem: given α,

find S that maximizes $\alpha f(S) - c(S)$

$$\Leftrightarrow$$

find S that maximizes $f(S) - \frac{1}{\alpha} c(S)$

demand set with "prices" c/α

(in markets for goods)

Demand Set (in Markets for Goods)

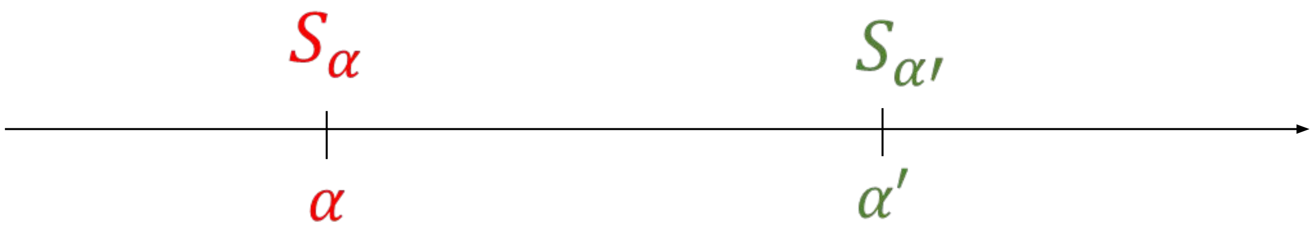
- Demand set $D(f, p) =$ a set S maximizing utility $\mathrm{u}(\mathrm{S}) = f(S) - \sum_{j \in S} p_j$
- Key property of gross substitutes - GREEDY algorithm solves the demand set problem (add element with maximal marginal utility)

  [Tie-breaking: high-cost, then low index, include actions with marginal utility 0]

Step 1: Next Critical α

- Demand set $D(f, p) =$ a set S maximizing utility $\mathrm{u(S)} = f(S) - \sum_{j \in S} p_j$

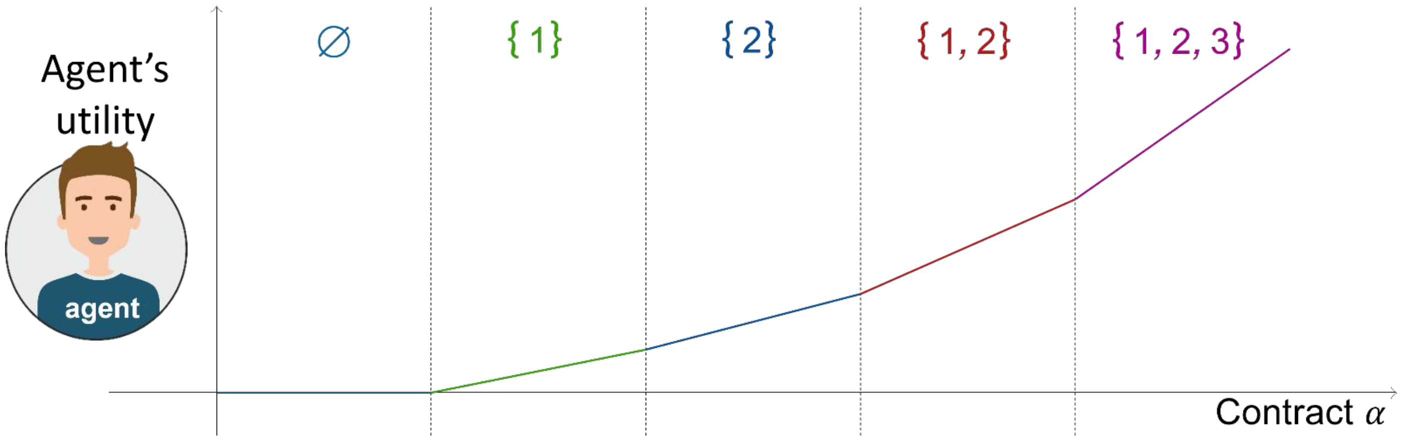
- Let $S_\alpha = (a_1, a_2, \ldots, a_d)$ and $S_{\alpha'}$ be respective demand sets of α, α'
- Either: $S_\alpha[i] \neq S_{\alpha'}[i]$ for some $i \leq d$, or $|S_{\alpha'}| > d$
- Suffices to consider finitely many potential values for α' (for each action and index), and take the smallest one that is larger than α



- Generic cost: at most one iteration in greedy in which tie breaking occurs (in particular, c(a1)\neq c(a2)).
- For generic cost: at most n(n+1)/2 critical points
  - By showing that the potential of sum of ranks is increasing in critical alphas

  - Perturbations do not introduce new sets into the demand
  - Draw cost uniformly at random in [c,c+epsilon], new cost is generic with probability 1

Step 2: Poly-Many Critical α's



- **Lemma:** at each critical point:
  - an action is added to S, or
  - an action from S is replaced by one with higher cost

(obtained by perturbing cost, so that GREEDY has at most one tie-breaking)

The agent's problem: given α,

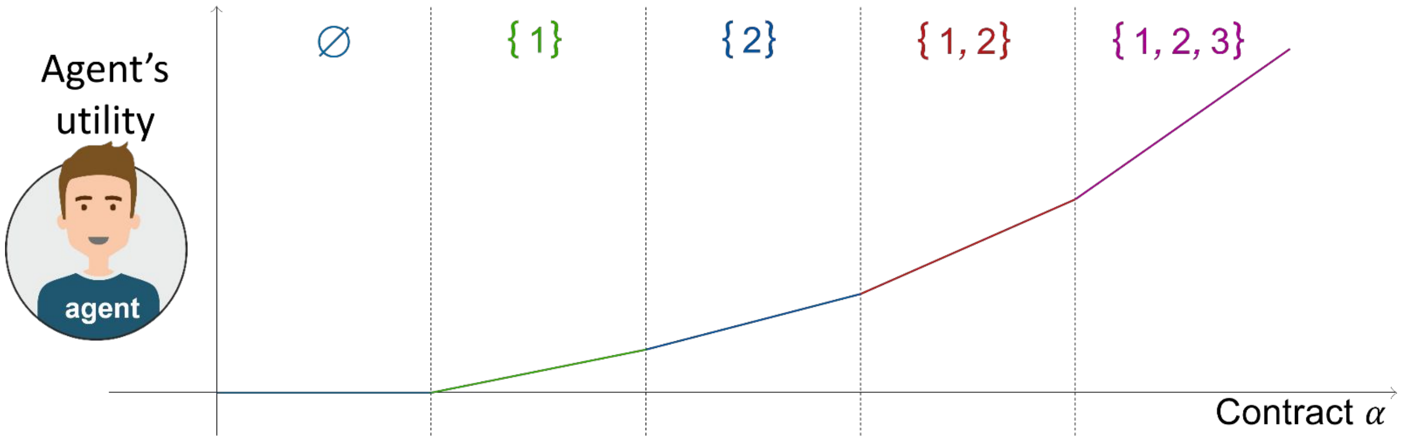
find S that maximizes $\alpha f(S) - c(S)$

$$\Leftrightarrow$$

find S that maximizes $f(S) - \frac{1}{\alpha} c(S)$

- Reorder actions: $c(a_1) < \cdots < c(a_n)$
- Define $\phi(a_i) = i, \phi(S) = \sum_{a \in S} \phi(a)$
- ϕ is an integer $\leq n(n+1)/2$, which increases at every critical α
- Conclusion: $O(n^2)$ critical points for GS
- (this is tight)

Proof Sketch: For GS $|C_{f,c}|$ is polynomial in n



The agent's problem: given α,

find S that maximizes $\alpha f(S) - c(S)$

$$\Leftrightarrow$$

find S that maximizes $f(S) - \frac{1}{\alpha} c(S)$

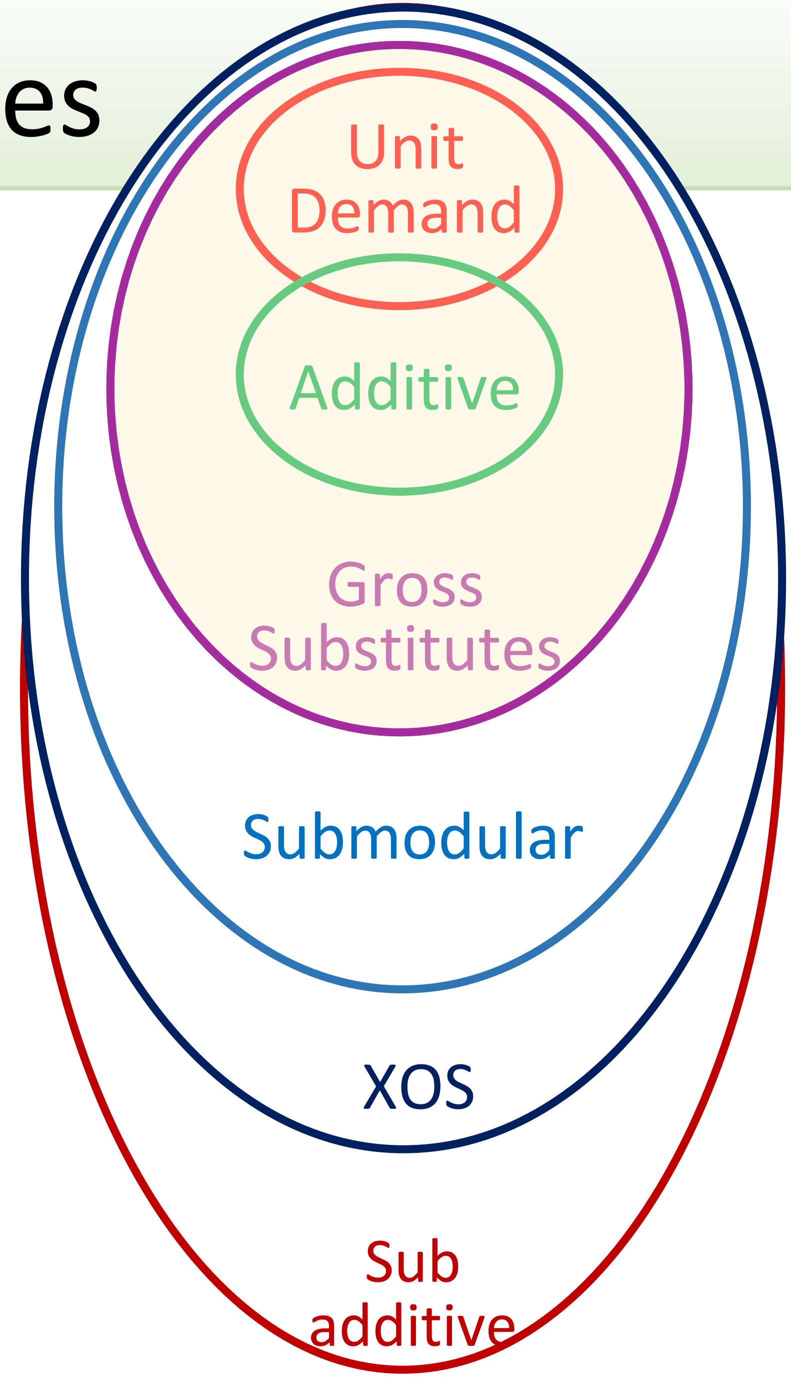
- This is precisely a demand query!
- **Non-standard:** All prices go down simultaneously, at rate $\frac{1}{\alpha}$
- **Theorem:** For GS functions, at each critical point:
  - an action is added to S, or
  - an action from S is replaced by one with higher cost
- Potential function argument showing that $|C_{f,c}| = O(n^2)$

Beyond Gross Substitutes

Submodular: $f(i \mid S) \geq f(i \mid T)$ for $S \subseteq T$
(decreasing marginal value)

XOS: maximum over additive
(also: fractionally subadditive)

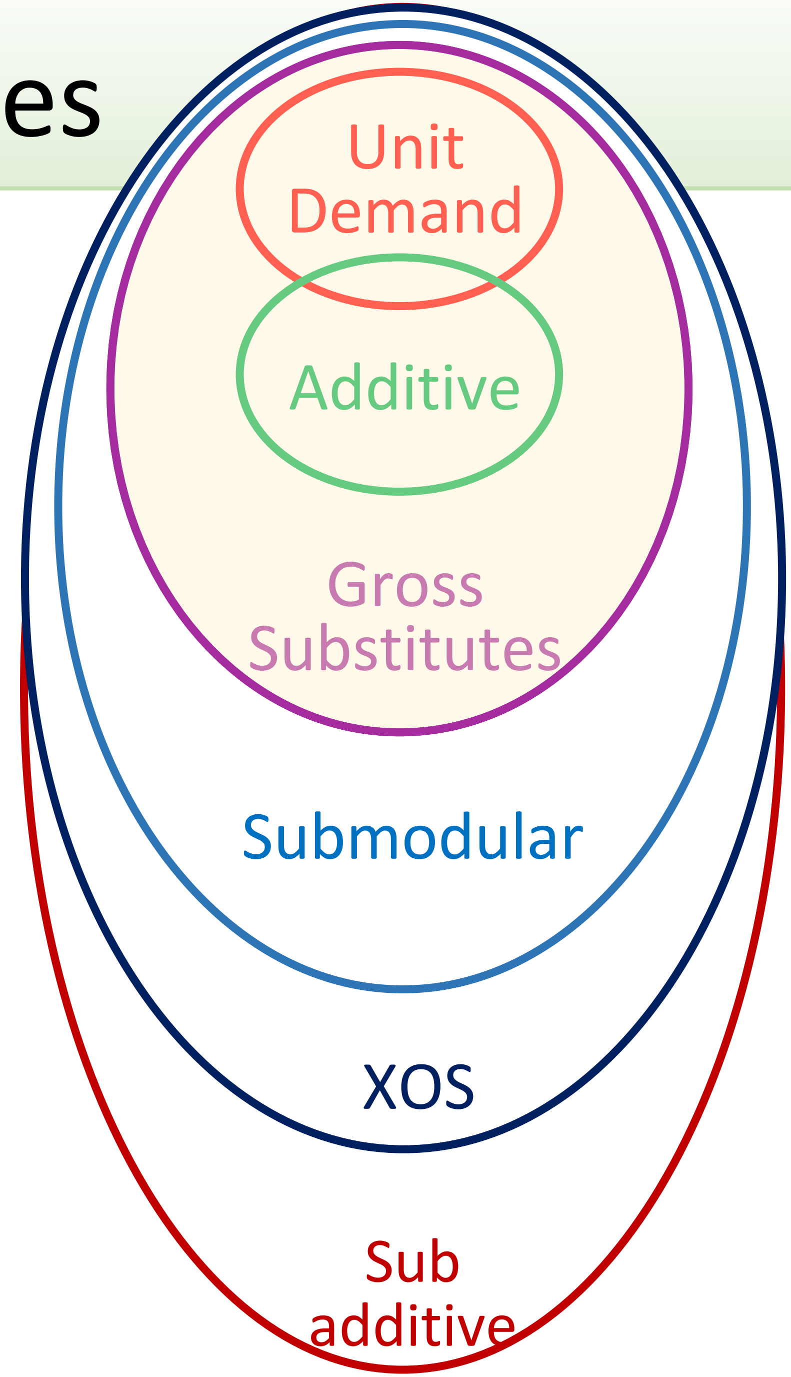
Subadditive: $f(S) + f(T) \geq f(S \cup T)$



Beyond Gross Substitutes

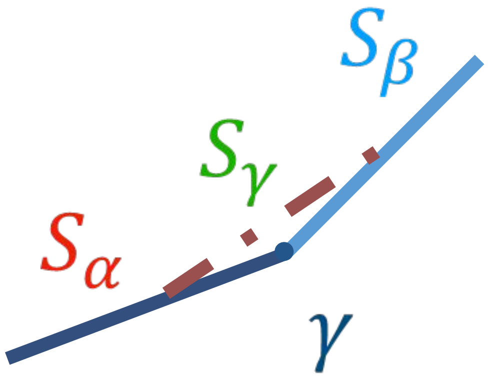
- Theorem [EDFK'21]: For submodular rewards:
  - exponentially many critical points
  - Optimal contract is NP-hard
- Inapproximability results [EFS'23]:
  - No PTAS for submodular rewards with value queries
  - No constant-approximation for XOS rewards with value queries

- Value query: Given S, return $f(S)$
- Demand query: Given action "prices" $p_1, \dots, p_n$, return S maximizing $f(S) - \sum_{i \in S} p_i$



Is Gross Substitutes a frontier?

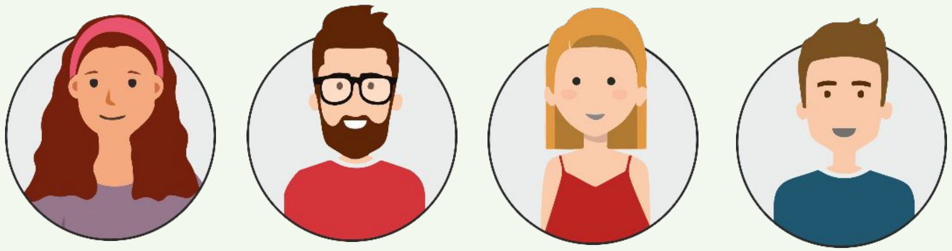
- Theorem [DFG'24]: For every f, c, a demand oracle (i.e., agent's BR) is sufficient for enumerating all critical values
- Proof idea: For a segment $[\alpha, \beta]$, use the oracle to get S_α and S_β.
  - If $S_\alpha = S_\beta$: the utility is linear in $[\alpha, \beta]$
  - Otherwise, query again at $\gamma = \frac{c(S_\alpha) - c(S_\beta)}{f(S_\alpha) - f(S_\beta)}$
  - If $c(S_\gamma) - c(S_\beta)$: the utility is linear in $[\alpha,\gamma)$ and in $[\gamma,\beta]$
  - Otherwise, there are more than 2 linear pieces; solve recursively for $[\alpha,\gamma]$ and $[\gamma,\beta]$



- Upshot: For every monotone f, c, a demand oracle and poly-many critical values are sufficient to find the optimal contract
- Corollary: an efficient algorithm for supermodular f and submodular c

Multiple Actions: Summary

- **Key take-aways:**
  - Gross substitutes is a ``frontier of tractability’’ for combinatorial contracts
  - (Perhaps) surprising connection to auctions
- **Additional results in the paper:**
  - FPTAS for general functions f, under access to demand oracle
  - Robust optimality of linear contracts for non-binary outcomes
  - Extension of computational results to linear contracts for non-binary outcomes
- **Open problems:**
  - Polynomial-time algorithm for submodular valuations with demand queries?
  - Extension to multiple agents



Multiple agents

[F, Chuang, Stoica, Shenker EC’05,
Babaioff F Nisan EC’06, Emek F ‘09,
Ezra Duetting F Kesselheim, STOC’23]

Combinatorial Agency Model

[Babaioff F Nisan 2006]

- n agents
- Binary action: $A_i = \{0,1\}$
  (0: no effort, 1: effort)
- Cost c_i: cost of effort (no effort = no cost)
- Binary outcome: {0,1}
- Principal receives reward 1 for success
- Success probability function $f: \{0,1\}^n \rightarrow [0,1]$

Contracts and Objective

- Optimal (=linear) contract: $\alpha = (\alpha_1, \ldots, \alpha_n)$
  $\alpha_i \geq 0$: payment to agent i for success

- **Agent's perspective:** Agent i prefers ``effort'' over ``no effort'' iff

$$\underbrace{\alpha_i f(S) - c_i}_{\text{agent } i\text{'s utility under effort}} \geq \underbrace{\alpha_i f(S - \{i\})}_{\text{agent } i\text{'s utility under no effort}}$$

$\Rightarrow \quad \alpha_i = \frac{c_i}{f(i \mid S-i)}$ is the best way to incentivize agent i

- Principagent's al's perspective: Find the set of agents S that maximizes

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

- Problem: compute optimal contract for submodular/XOS/subadditive f

- Challenge: even if f is highly structured, g is a mess

Contracts and Objective

- Optimal (=linear) contract: $\alpha = (\alpha_1, \ldots, \alpha_n)$
  $\alpha_i \geq 0$: payment to agent i for success

- **Agent's perspective:** Agent i prefers ``effort'' over ``no effort'' iff

$$\alpha_i f(S) - c_i \geq \alpha_i f(S - \{i\})$$

$\Rightarrow \quad \alpha_i = \frac{c_i}{f(i \mid S-i)}$ is the best way to incentivize agent i

- Principagent's al's perspective: Find the set of agents S that maximizes

$$g(S) = f(S)(1 - \textstyle\sum_{i \in S} \frac{c_i}{f(i \mid S-i)})$$

- Problem: compute optimal contract for submodular/XOS/subadditive f

- Challenge: even if f is highly structured, g is a mess

Contracts and Objective

- Optimal (=linear) contract: $\alpha = (\alpha_1, \dots, \alpha_n)$
  $\alpha_i \geq 0$: payment to

- **Agent's perspective** t'' over ``no effort'' iff

$$\alpha_i \quad - \{i\})$$

$$\Rightarrow \quad \alpha_i = \frac{c_i}{f(i \mid S-i)}$$ is the best way to incentivize agent i

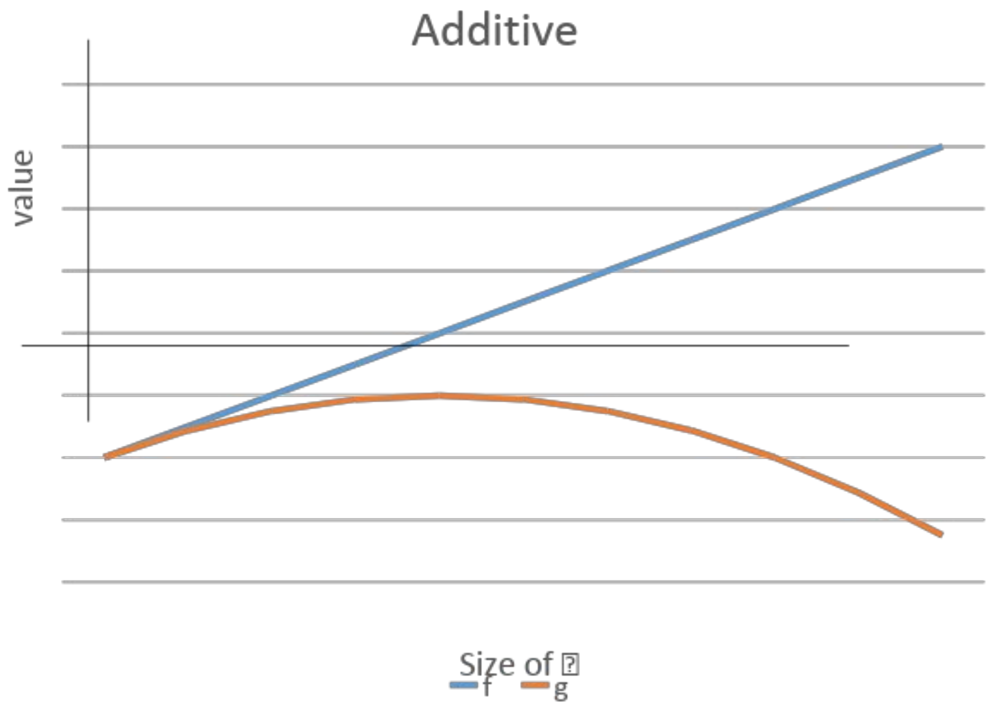
"margin" of i w.r.t. S:
$f(i \mid S-i)$
$= f(S) - f(S-i)$

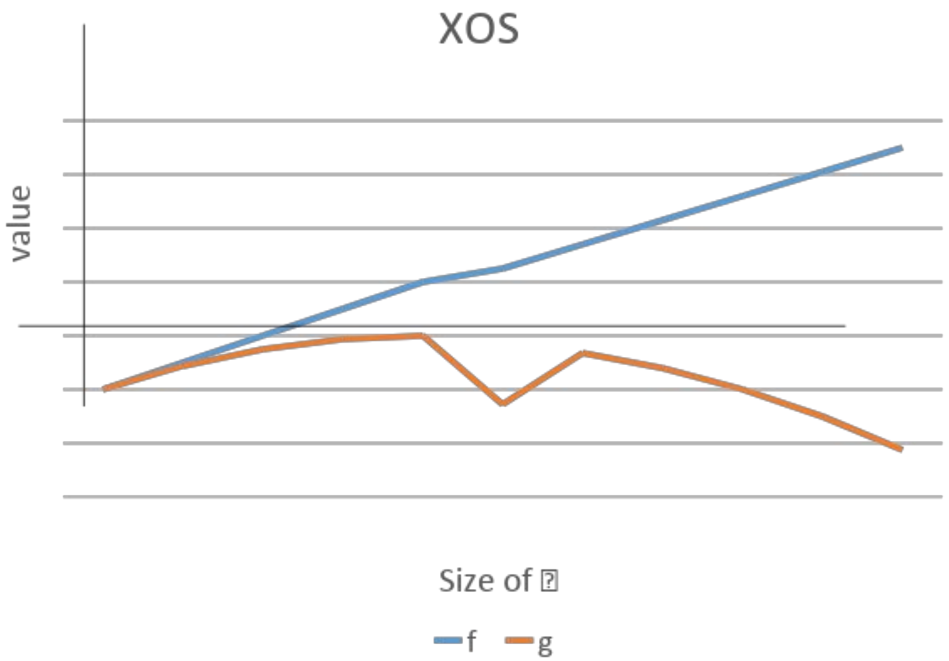
- **Principal's perspective:** Find the set of agents S that maximizes

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

- **Problem**: compute optimal contract for submodular/XOS/subadditive f
- **Challenge**: even if f is highly structured, g is highly non-structured

Contracts and Objective





Warmup: Additive f

Theorem: The problem is NP-hard even for additive f, but admits an FPTAS

Proof: via reduction from Partition:

- Partition: given $\{w_1, \dots, w_n\}$ of positive integers summing to W, can it be partitioned into two sets that sum to $W/2$ each?
- Contract instance: additive f, every agent has value w_i and cost $c_i = \frac{w_i^2}{W}$
- To incentivize agent i: $\alpha_i = w_i/W$
- Principal's utility: $g(S) = \sum_{i \in S} w_i \cdot \left(1 - \sum_{i \in S} \frac{w_i}{W}\right)$
  - Maximized when $\sum_{i \in S} w_i = W/2$
  - Thus, Partition is solvable iff $g(S^*) = W/4$

Warmup: Additive f

Theorem: The problem is NP-hard even for additive f, but admits an FPTAS

Proof: via reduction from PARTITION

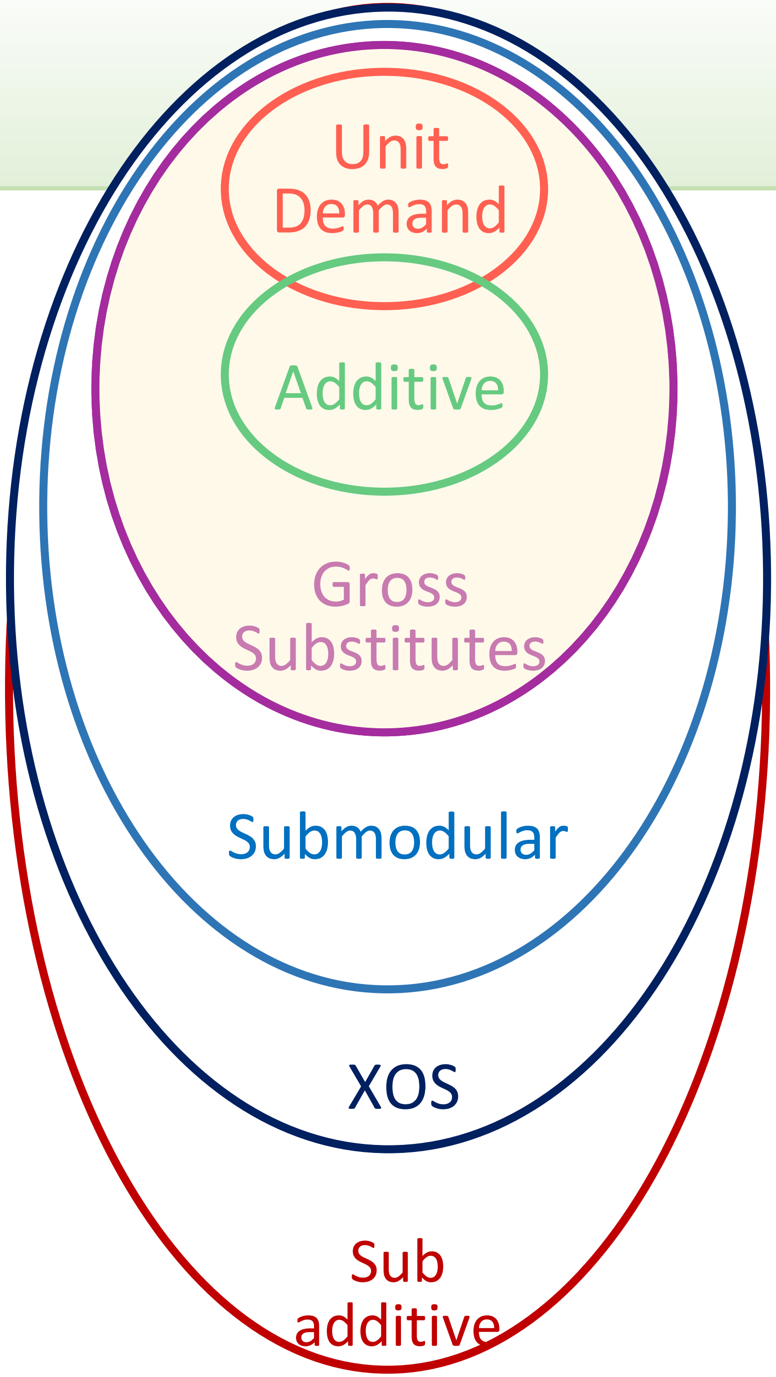
- PARTITION: given a multiset of integers that sum to W, determine whether one can partition them into to sets that sum to $W/2$
- Construct a contract instance (i.e., $\{f_i\}, \{c_i\}$) where the principal's utility is maximized when the sum of agent values sum to $W/2$

Submodular/XOS/Subadditive f

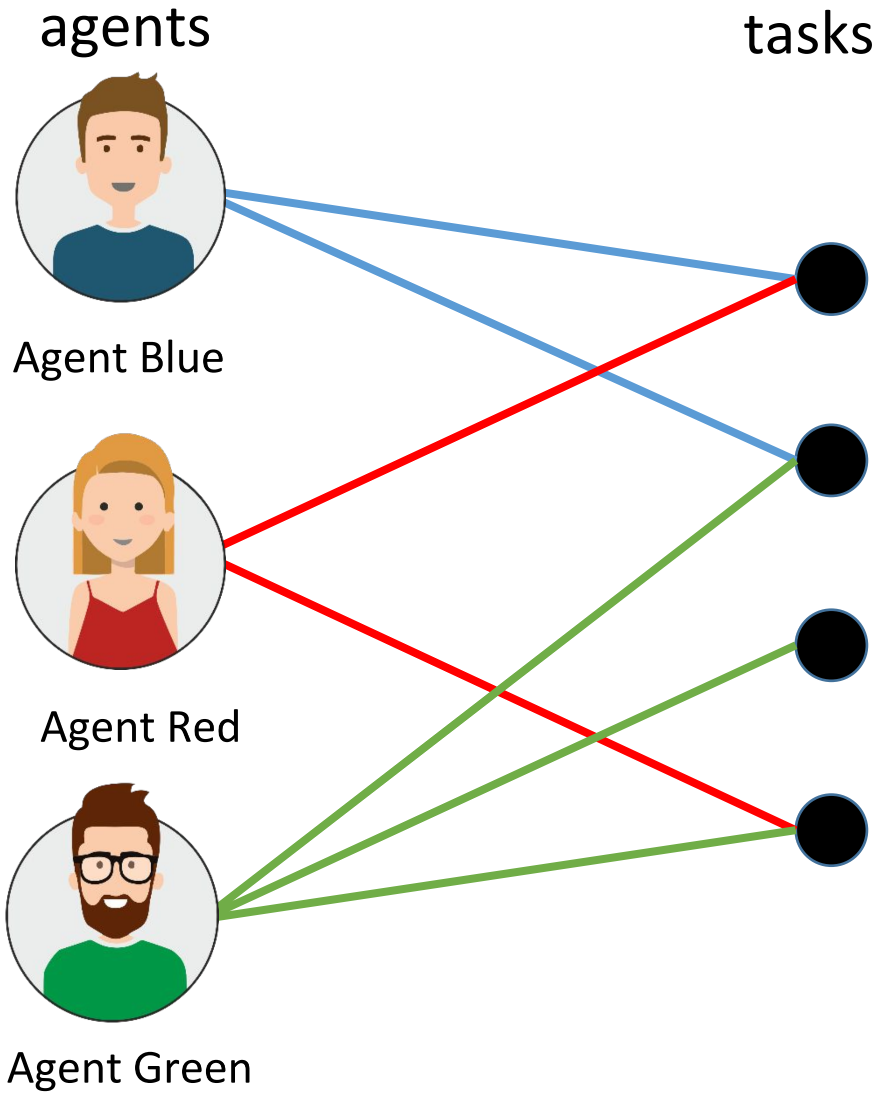
Submodular: $f(i \mid S) \geq f(i \mid T)$ for $S \subseteq T$
(decreasing marginal value)

XOS: maximum over additive
(also: fractionally subadditive)

Subadditive: $f(S) + f(T) \geq f(S \cup T)$



Unweighted Coverage Function (submodular)



agents

tasks

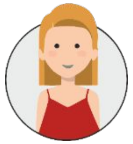
Agent Blue

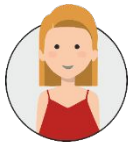
Agent Red

Agent Green

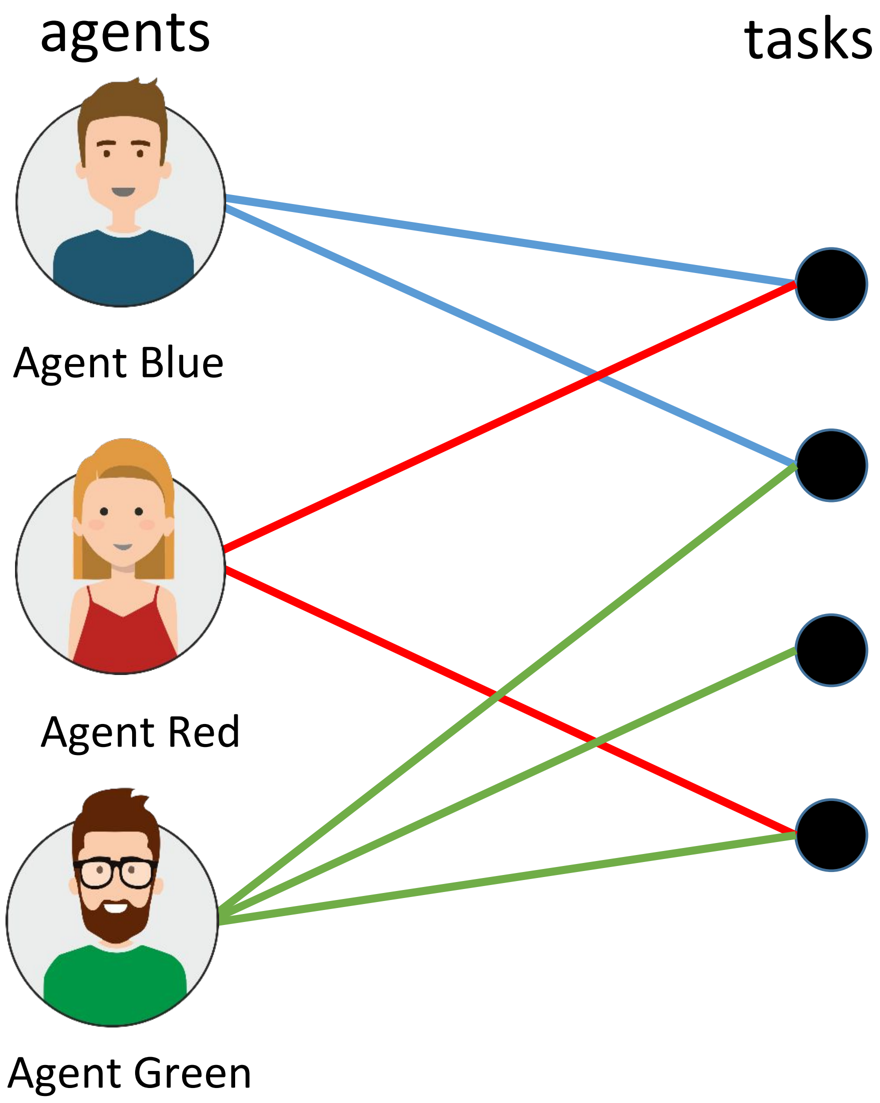
f(set of agents) $=$
# tasks covered by these agents

e.g.:

$f($  $) = 2$

$f($  $|$ $) = 1$

Unweighted Coverage Function (submodular)



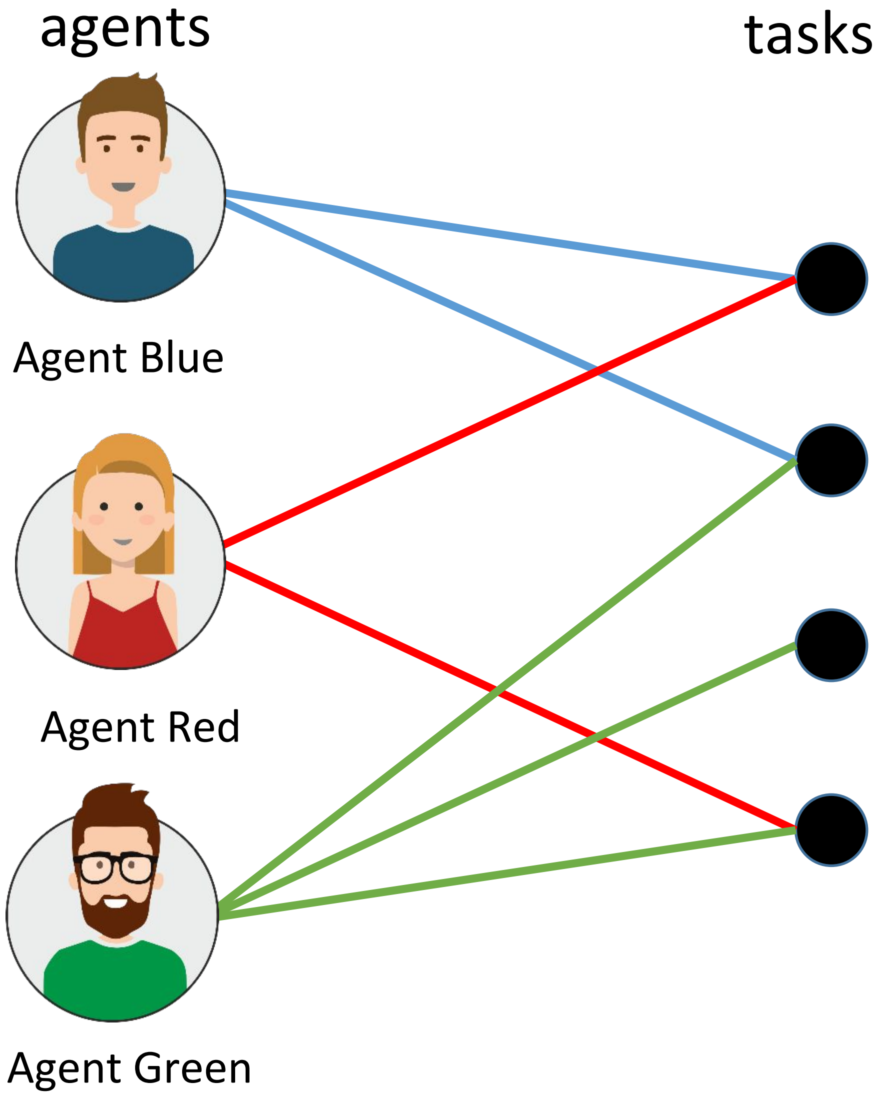
Principal's objective:

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

Total # tasks covered by S

tasks covered uniquely by agent i

Unweighted Coverage Function (submodular)



Principal’s objective:

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

Total # tasks covered by S

tasks covered uniquely by agent i

Unique coverage is hard to approximate within a constant factor [Demaine Feige Hajiaghayi Salavatipour 2006]

Approximation Results for Submodular/XOS/Subadditive

[Dutting Ezra Feldman Kesselheim, STOC'23]

Results:

- **(+)** There is a polynomial-time algorithm for finding a $O(1)$-approximate contract for submodular f, using value oracle, and for XOS f, using demand oracle
- **(-)** No better than constant-approximation for XOS f, using demand and value oracles
- **(-)** No better than $\Omega(\sqrt{n})$-approximation for subadditive f, using demand and value oracles (even for f constant close to submodular)

Approximation Results for Submodular/XOS/Subadditive

Main Results:

- **(+)** There is a polynomial-time algorithm for finding a $O(1)$-approximate contract for submodular f, using value oracle, and for XOS f, using demand oracle

Approximation Results for Submodular/XOS/Subadditive

Main Results:

- **(+)** There is a polynomial-time algorithm for finding a $O(1)$-approximate contract for submodular f, using value oracle, and for XOS f, using demand oracle
- **(-)** No better than $\Omega(\sqrt{n})$-approximation for subadditive f, using demand and value oracles (even for f constant close to submodular)

- No better than constant-approximation for XOS f, using demand and value oracles
- For additive it is NP-hard to find the optimal contract, but there is a FPTAS

$g(S) = f(S)(1 - \sum_{i \in S} \frac{c_i}{f(i \mid S_{-i})})$

Proof Sketch (XOS)

- **Goal**: find S satisfying $g(S) \geq \text{const} \cdot g(S^*)$, where S^* is optimal set
- Assume: (i) $f(S^*)$ is known, (ii) individual contributions are negligible
- **Lemma 1**: $\sum_{i \in S^*} \sqrt{c_i} \leq \sqrt{f(S^*)}$
- **Lemma 2**: If for all $i \in S$, $f(\, i \mid S \setminus \mathrm{i}\,) \geq \sqrt{2 c_i f(S)}$, then $g(S) \geq \frac{1}{2} f(S)$
- Non-conventional use of demand queries:
  - Define a “price” $p_i = \frac{1}{2}\sqrt{c_i f(S^*)}$, and consider “demand set” T (maximizing $f(S) - \sum_{i \in S} p_i$)
  - $f(T) \geq f(T) - \sum_{i \in T} p_i \geq f(S^*) - \sum_{i \in S^*} p_i \geq \frac{1}{2} f(S^*)$ [by def. of demand and Lemma 1]
  - Also: $f(\, i \mid T \setminus i\,) \geq p_i = \frac{1}{2}\sqrt{c_i f(S^*)}$ we wish this to be $\geq \sqrt{2 c_i f(T)}$ to use Lemma 2
- A novel scaling property of XOS: can scale down value of $f(T)$ to $f(U)$ by removing items, and keeping the marginals of remaining items large
- **Altogether we get**: $g(U) \geq \frac{1}{2} f(U) \geq \text{const} \cdot f(T) \geq \text{const} \cdot f(S^*) \geq \text{const} \cdot g(S^*)$ ■

$g(S) = f(S)(1 - \sum_{i \in S} \frac{c_i}{f(i \mid S_{-i})})$

Proof Sketch (XOS)

- **Goal**: find S satisfying $g(S) \geq \text{const} \cdot g(S^*)$, where S^* is optimal set
- Let T be the demand set under prices $p_i = \frac{1}{2}\sqrt{c_i f(S^*)}$
- **Lemma 1**: $f(T) \geq \frac{1}{2} f(S^*)$ [so we can get a set that approximates $f(S^*)$]
- **Lemma 2**: For every set S, if $f(i \mid S - \mathrm{i}) \geq \sqrt{2 c_i f(S)}$ for all $i \in S$, then $g(S) \geq \frac{1}{2} f(S)$ (so, sufficient to approximate f, instead of messy g)
- Since T is a demand set, $f(i \mid T - i) \geq p_i = \frac{1}{2}\sqrt{c_i f(S^*)}$

$g(S) = f(S)(1 - \sum_{i \in S} \frac{c_i}{f(i \mid S_{-i})})$

Proof Sketch (XOS)

- **Goal**: find S satisfying $g(S) \geq \text{const} \cdot g(S^*)$, where S^* is optimal set
- Let T be the demand set under prices $p_i = \frac{1}{2}\sqrt{c_i f(S^*)}$
- **Lemma 1**: $f(T) \geq \frac{1}{2} f(S^*)$ [so we can get a set that approximates $f(S^*)$]
- **Lemma 2**: For every set S, if $f(i \mid S - \mathrm{i}) \geq \sqrt{2c_i f(S)}$ for all $i \in S$, then $g(S) \geq \frac{1}{2} f(S)$ (so, sufficient to approximate f, instead of messy g)
- Since T is a demand set, $f(i \mid T - i) \geq p_i = \frac{1}{2}\sqrt{c_i f(S^*)} \geq \sqrt{2c_i f(T)}$ ← Desired (to use Lemma 2)
  - **Problem**: $f(T)$ may be too large
  - **Idea**: remove agents from T until inequality is satisfied
  - **Problem**: marginals may decrease (unlike submodular)
- **Thm**: a novel scaling property of XOS: scale down $f(T)$ and keep marginals high enough
- **Altogether**: $g(S) \geq \frac{1}{2} f(S) \geq \text{const} \cdot f(T) \geq \text{const} \cdot f(S^*) \geq \text{const} \cdot g(S^*)$ ■

Two Birds with One Stone: Scaling Property of XOS

Key Lemma: Given a set T and a parameter $x \leq f(T)$
it is possible to find in polytime using value queries a set $U \subseteq T$ such that:

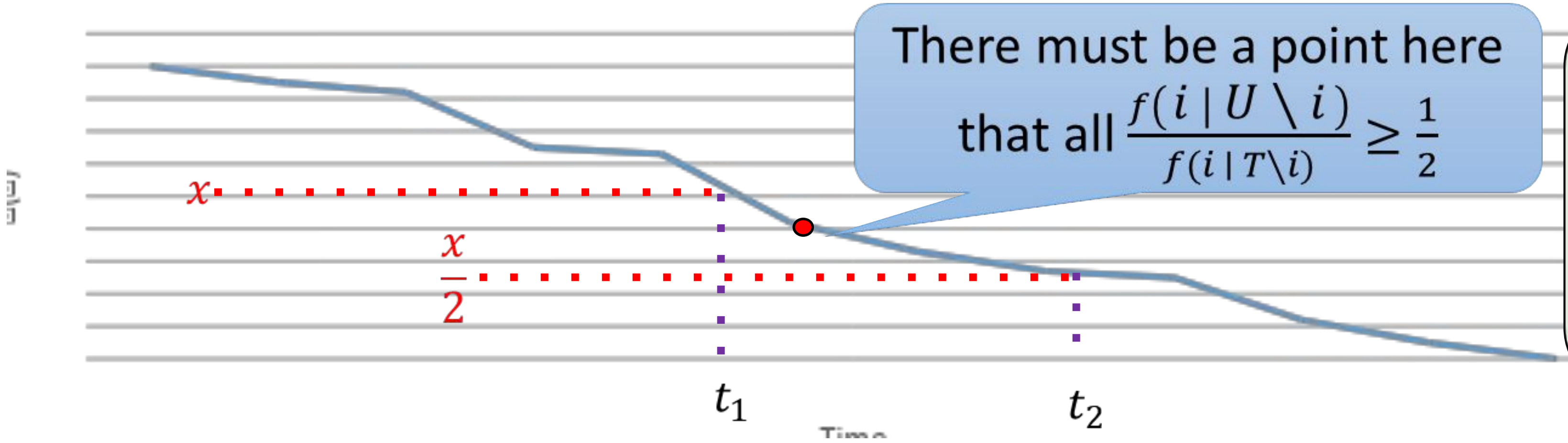
(1) $\frac{x}{2} \leq f(U) \lesssim x$

(2) $f(i \mid U \setminus i) \geq \frac{f(i \mid T \setminus i)}{2}$ for all $i \in U$

Up to one agent

Algorithm: Start with $U = T$

Each turn, delete the agent with the minimal value of $\frac{f(i \mid U \setminus i)}{f(i \mid T \setminus i)}$



There must be a point here that all $\frac{f(i \mid U \setminus i)}{f(i \mid T \setminus i)} \geq \frac{1}{2}$

$$\sum_{i=t_1}^{t_2} f(i \mid U \setminus i) \approx \frac{x}{2}$$

$$\sum_{i=t_1}^{t_2} f(i \mid T \setminus i) \approx x$$

XOS

Constant Approximation for XOS

Theorem [Dutting Ezra Feldman Kesselheim, working paper]

There is a polynomial-time algorithm for finding a $O(1)$-approximate contract for XOS f.

(assuming demand oracle access to f)

How? Novel relaxation of OPT that exploits local-optimality criteria, prices derived from that + approximated demand query

Also: For f subadditive, may need exponentially many value queries to get $O(\sqrt{n})$ approx.

Key Insight: Novel Relaxation

- Consider relaxation, for parameter $0 < \gamma < 1$:

$$\begin{array}{ll} \text{maximize} & f(S^{**}) \\ \text{subject to} & (f(i \mid S^{**} - \{i\}))^2 \geq c_i f(S^{**})/\gamma \end{array}$$

Lemma [D., Ezra, Feldman, Kesselheim 2022]

If for the optimal set of agents S^* and some $0 < \varepsilon < \gamma$, $f(i) \leq \varepsilon f(S^*)$ for all $i \in S^*$, then for the optimal solution S^{**} to the relaxation it holds that

$$f(S^{**}) \geq (\gamma - \varepsilon) f(S^*).$$

From Relaxation to Contract (Sketch)

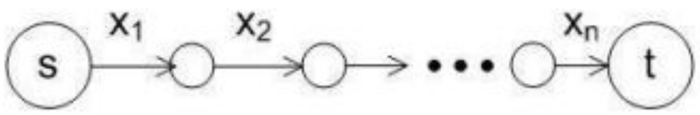
- Suppose we know $f(S^{**})$ (just the objective value, not the set itself)

- Let $\Delta_i = \sqrt{c_i\, f(S^{**})/\gamma}$ and run the following algorithm:
  1. Post prices $\Delta_i/2$
  2. $T \leftarrow$ approximate demand query at these prices
  3. U maximal subset of T s.t. $\sum_{i\,\in\,U} f(\,i \mid T\; - \{i\}\,) \leq f(S^{**})$

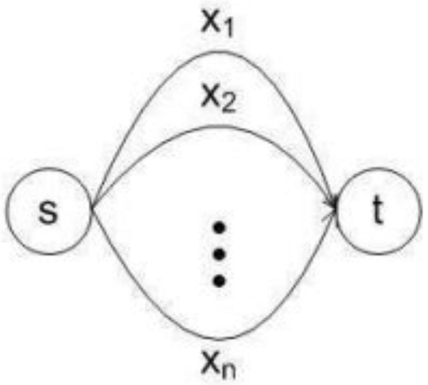
- Argue that if $f(i) \leq\; \varepsilon f(S^{*})$ for all $i\; \in S^{*}$ then this yields an O(1)-approximation

How this Fits into Known Results

- Babaioff F Nisan (2006):
  - For general f: exp. many value queries
  - For f encoded as read-once network: #P-complete
  - Poly-time algorithm for AND read-once networks
  - Conjecture: Polynomial-time algorithm for series-parallel read-once networks
- F Emek (2009):
  - NP-hard + FPTAS for OR read-once network
  - "Almost FPTAS" for series-parallel read once networks



(a) AND technology



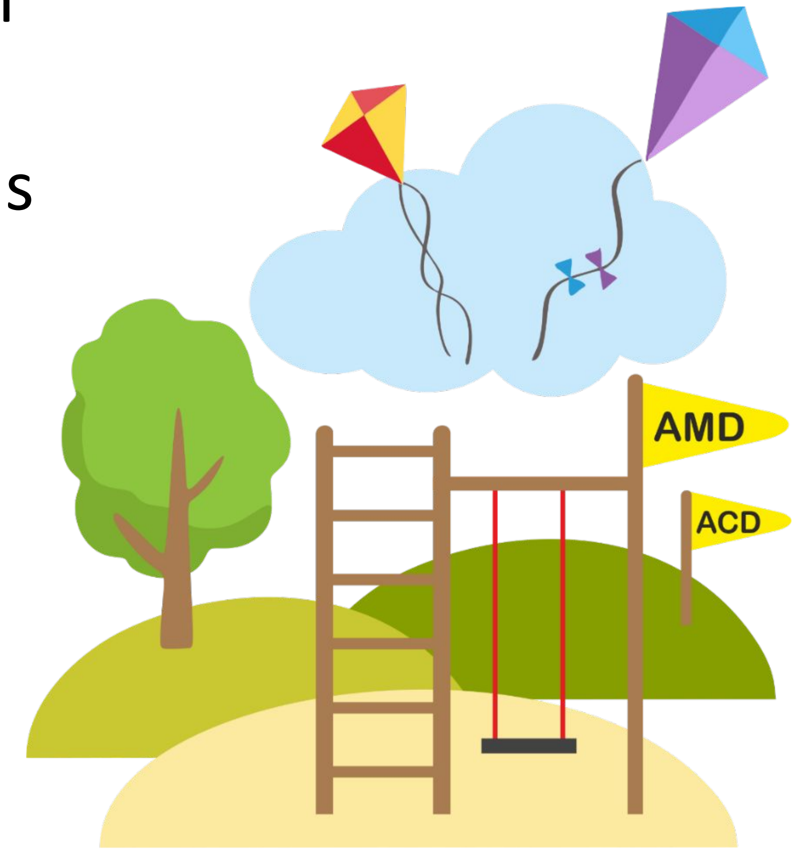
(b) OR technology

Multiple Agents: Summary

- **Key take-aways:**
  - First constant-factor approximation for a contract problem
  - Non-standard use of demand queries
  - New scaling property of XOS functions, that may be of independent interest
- **Open problems:**
  - Beyond binary actions? [some progress in a working paper]
  - Approximation algorithm for general series-parallel graphs?

Main Take Aways

- Contract theory is a new frontier in AGT
- Complexity and approximation shed new light on contract design
- Interesting connections to combinatorial auctions and other combinatorial optimization problems
    - E.g., gross substitutes as tractability frontier
- Many fundamental problems still open



Thank You!