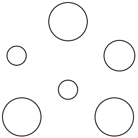
Generalized coverage problems: approximation through game design

Dario Paccagnan

Joint work with J. R. Marden (UCSB)

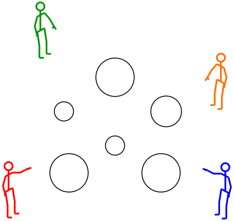
Combinatorial resource allocation

- a set of resources



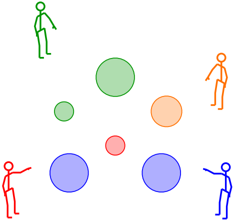
Combinatorial resource allocation

- a set of resources
- a set of agents



Combinatorial resource allocation

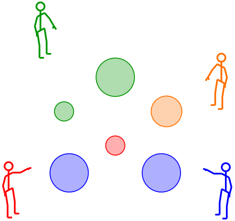
- a set of resources
- a set of agents



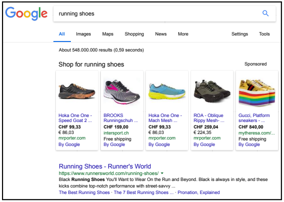
Goal: assign resources to agents to maximize a given welfare function

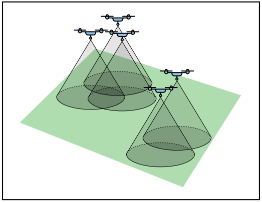
Combinatorial resource allocation

- a set of resources
- a set of agents



Goal: assign resources to agents to maximize a given welfare function

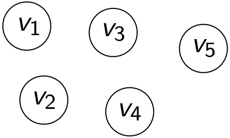




Generalized Multiagent Maximum Coverage (GMMC)

Generalized Multiagent Maximum Coverage (GMMC)

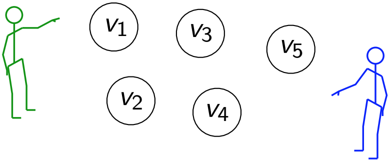
resources: $r \in \mathcal{R}, \quad v_r \geq 0$



Generalized Multiagent Maximum Coverage (GMMC)

resources: $r \in \mathcal{R}, \quad v_r \geq 0$

agents: $i \in \{1, \ldots, n\}$

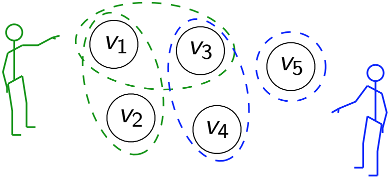


Generalized Multiagent Maximum Coverage (GMMC)

resources: $r \in \mathcal{R}, \quad v_r \geq 0$

agents: $i \in \{1, \ldots, n\}$

allocations: $a_i \in \mathcal{A}_i \subseteq 2^{\mathcal{R}}$

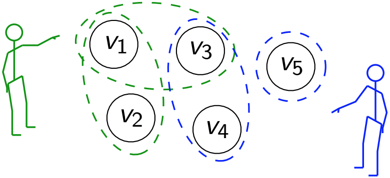


Generalized Multiagent Maximum Coverage (GMMC)

resources: $r \in \mathcal{R}, \quad v_r \geq 0$

agents: $i \in \{1, \ldots, n\}$

allocations: $a_i \in \mathcal{A}_i \subseteq 2^{\mathcal{R}}$



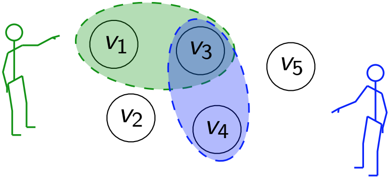
welfare: $W(a) = \sum_{r \in \cup_i a_i} v_r w_r(|a|_r)$ $\qquad w_r : \mathbb{N} \to \mathbb{R}_{\geq 0}$

Generalized Multiagent Maximum Coverage (GMMC)

resources: $r \in \mathcal{R}, \quad v_r \geq 0$

agents: $i \in \{1, \ldots, n\}$

allocations: $a_i \in \mathcal{A}_i \subseteq 2^{\mathcal{R}}$



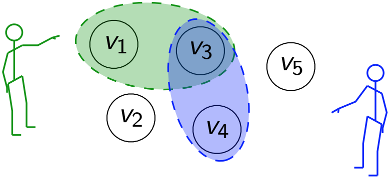
welfare: $W(a) = \sum_{r \in \cup_i a_i} v_r w_r(|a|_r)$ $\qquad w_r : \mathbb{N} \to \mathbb{R}_{\geq 0}$

Generalized Multiagent Maximum Coverage (GMMC)

resources: $r \in \mathcal{R}, \quad v_r \geq 0$

agents: $i \in \{1, \ldots, n\}$

allocations: $a_i \in \mathcal{A}_i \subseteq 2^{\mathcal{R}}$



welfare: $W(a) = \sum_{r \in \cup_i a_i} v_r w_r(|a|_r) \qquad w_r : \mathbb{N} \to \mathbb{R}_{\geq 0}$

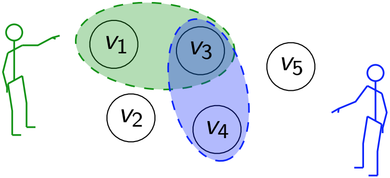
System-level objective: $\max_{a \in \mathcal{A}} W(a)$

Generalized Multiagent Maximum Coverage (GMMC)

resources: $r \in \mathcal{R}, \quad v_r \geq 0$

agents: $i \in \{1, \ldots, n\}$

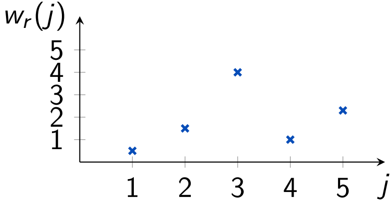
allocations: $a_i \in \mathcal{A}_i \subseteq 2^{\mathcal{R}}$



welfare: $W(a) = \sum_{r \in \cup_i a_i} v_r w_r(|a|_r)$ $\qquad w_r : \mathbb{N} \to \mathbb{R}_{\geq 0}$

$$\boxed{\text{System-level objective: } \max_{a \in \mathcal{A}} W(a)}$$

▷ no constraints on $w_r(j)$

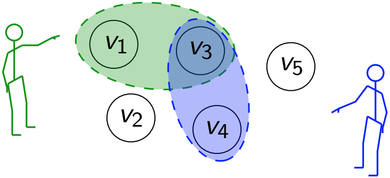


Generalized Multiagent Maximum Coverage (GMMC)

resources: $r \in \mathcal{R}, \quad v_r \geq 0$

agents: $i \in \{1, \ldots, n\}$

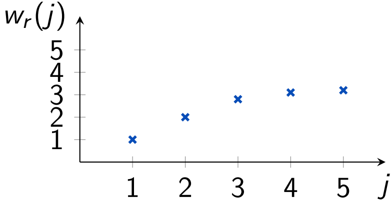
allocations: $a_i \in \mathcal{A}_i \subseteq 2^{\mathcal{R}}$



welfare: $W(a) = \sum_{r \in \cup_i a_i} v_r w_r(|a|_r)$ $\qquad w_r : \mathbb{N} \to \mathbb{R}_{\geq 0}$

$\boxed{\text{System-level objective: } \max_{a \in \mathcal{A}} W(a)}$

▷ no constraints on $w_r(j)$
typically concave

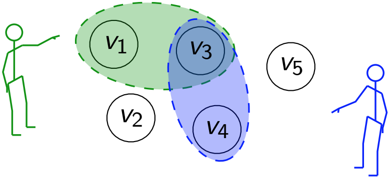


Generalized Multiagent Maximum Coverage (GMMC)

resources: $r \in \mathcal{R}, \quad v_r \geq 0$

agents: $i \in \{1, \ldots, n\}$

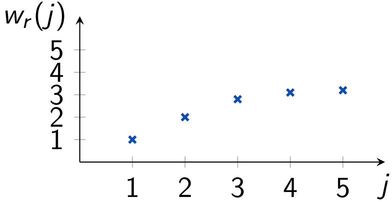
allocations: $a_i \in \mathcal{A}_i \subseteq 2^{\mathcal{R}}$



welfare: $W(a) = \sum_{r \in \cup_i a_i} v_r w_r(|a|_r)$ $\qquad w_r : \mathbb{N} \to \mathbb{R}_{\geq 0}$

$$\boxed{\text{System-level objective: } \max_{a \in \mathcal{A}} W(a)}$$

- ▷ no constraints on $w_r(j)$ typically concave
- ▷ to ease the presentation $w_r(j) = w(j)$



Connection with Coverage problems

Connection with Coverage problems

GMMC problem

Connection with Coverage problems

GMMC problem

- set of weighted resources: $\mathcal{R}$

Connection with Coverage problems

GMMC problem

- set of weighted resources: $\mathcal{R}$
- n collections of sets: $\{\mathcal{A}_i\}_{i=1}^{n} \subseteq 2^{\mathcal{R}}$

Connection with Coverage problems

GMMC problem

- set of weighted resources: $\mathcal{R}$
- n collections of sets: $\{\mathcal{A}_i\}_{i=1}^{n} \subseteq 2^{\mathcal{R}}$
- choose one set per collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r w(|a|_r)$$

Connection with Coverage problems

GMMC problem

- set of weighted resources: $\mathcal{R}$
- n collections of sets: $\{\mathcal{A}_i\}_{i=1}^n \subseteq 2^{\mathcal{R}}$
- choose one set per collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r w(|a|_r)$$

Max-n-cover

Connection with Coverage problems

GMMC problem

- set of weighted resources: $\mathcal{R}$

- n collections of sets: $\{\mathcal{A}_i\}_{i=1}^n \subseteq 2^{\mathcal{R}}$

- choose one set per collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r w(|a|_r)$$

Max-n-cover

- set of weighted resources: $\mathcal{R}$

Connection with Coverage problems

GMMC problem

- set of weighted resources: $\mathcal{R}$
- n collections of sets: $\{\mathcal{A}_i\}_{i=1}^n \subseteq 2^{\mathcal{R}}$
- choose one set per collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r w(|a|_r)$$

Max-n-cover

- set of weighted resources: $\mathcal{R}$
- *one* collection of sets: $\bar{\mathcal{A}} \subseteq 2^{\mathcal{R}}$

Connection with Coverage problems

GMMC problem

- set of weighted resources: $\mathcal{R}$
- n collections of sets: $\{\mathcal{A}_i\}_{i=1}^n \subseteq 2^{\mathcal{R}}$
- choose one set per collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r w(|a|_r)$$

Max-n-cover

- set of weighted resources: $\mathcal{R}$
- *one* collection of sets: $\bar{\mathcal{A}} \subseteq 2^{\mathcal{R}}$
- choose n sets from collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r$$

Connection with Coverage problems

GMMC problem
- set of weighted resources: $\mathcal{R}$
- *n* collections of sets: $\{\mathcal{A}_i\}_{i=1}^n \subseteq 2^{\mathcal{R}}$
- choose one set per collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r w(|a|_r)$$

Max-n-cover
- set of weighted resources: $\mathcal{R}$
- *one* collection of sets: $\bar{\mathcal{A}} \subseteq 2^{\mathcal{R}}$
- choose *n* sets from collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r$$

▷ GMMC subsumes max-n-cover (set $w(j) \equiv 1$, $\mathcal{A}_i = \mathcal{A}_j$ for all i, j)

Connection with Coverage problems

GMMC problem

- set of weighted resources: $\mathcal{R}$
- n collections of sets: $\{\mathcal{A}_i\}_{i=1}^n \subseteq 2^{\mathcal{R}}$
- choose one set per collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r w(|a|_r)$$

Max-n-cover

- set of weighted resources: $\mathcal{R}$
- *one* collection of sets: $\bar{\mathcal{A}} \subseteq 2^{\mathcal{R}}$
- choose n sets from collection to max

$$W(a) = \sum_{r \in \cup_i a_i} v_r$$

▷ GMMC subsumes max-n-cover (set $w(j) \equiv 1$, $\mathcal{A}_i = \mathcal{A}_j$ for all i, j)

▷ GMMC subsumes [Che04],[Gair09] (set $w(j) \equiv 1$)

[Che04] C. Chekuri et al. "Maximum Coverage Problem with Group Budget Constraints and Applications", APPROX 04

[Gair09] M. Gairing, "Covering Games: Approximation through Non-cooperation", WINE 09

Facts: hardness and approximability

Facts: hardness and approximability

max-n-cover problem:

Facts: hardness and approximability

max-n-cover problem:

- ▷ $\mathcal{NP}$-hard

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$
- Poly-algorithms achieve $1 - 1/e$

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$
- Poly-algorithms achieve $1 - 1/e$

GMMC problem:

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$
- Poly-algorithms achieve $1 - 1/e$

GMMC problem:

- $\mathcal{NP}$-hard

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$
- Poly-algorithms achieve $1 - 1/e$

GMMC problem:

- $\mathcal{NP}$-hard
- If w is concave and $\mathcal{A}_i = \mathcal{A}_j$, best poly-algorithm achieves $1 - c/e$ and is centralized, $c = 1 - (w(n) - w(n-1))$

[Svir17] M. Sviridenko, J. Vondrák, J. Ward, "Optimal Approximation for Submodular and Supermodular Optimization with Bounded Curvature", MOR 17

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$
- Poly-algorithms achieve $1 - 1/e$

GMMC problem:

- $\mathcal{NP}$-hard
- If w is concave and $\mathcal{A}_i = \mathcal{A}_j$, best poly-algorithm achieves $1 - c/e$ and is centralized, $c = 1 - (w(n) - w(n-1))$

Issues:

[Svir17] M. Sviridenko, J. Vondrák, J. Ward, "Optimal Approximation for Submodular and Supermodular Optimization with Bounded Curvature", MOR 17

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$
- Poly-algorithms achieve $1 - 1/e$

GMMC problem:

- $\mathcal{NP}$-hard
- If w is concave and $\mathcal{A}_i = \mathcal{A}_j$, best poly-algorithm achieves $1 - c/e$ and is centralized, $c = 1 - (w(n) - w(n-1))$

Issues:

- distributedness?

[Svir17] M. Sviridenko, J. Vondrák, J. Ward, "Optimal Approximation for Submodular and Supermodular Optimization with Bounded Curvature", MOR 17

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$
- Poly-algorithms achieve $1 - 1/e$

GMMC problem:

- $\mathcal{NP}$-hard
- If w is concave and $\mathcal{A}_i = \mathcal{A}_j$, best poly-algorithm achieves $1 - c/e$ and is centralized, $c = 1 - (w(n) - w(n-1))$

Issues:

- distributedness?
- best possible approximation?

[Svir17] M. Sviridenko, J. Vondrák, J. Ward, "Optimal Approximation for Submodular and Supermodular Optimization with Bounded Curvature", MOR 17

Facts: hardness and approximability

max-n-cover problem:

- $\mathcal{NP}$-hard
- $\mathcal{NP}$-hard to approximate within any ratio better than $1 - 1/e$
- Poly-algorithms achieve $1 - 1/e$

GMMC problem:

- $\mathcal{NP}$-hard
- If w is concave and $\mathcal{A}_i = \mathcal{A}_j$, best poly-algorithm achieves $1 - c/e$ and is centralized, $c = 1 - (w(n) - w(n-1))$

Issues:

- distributedness?
- best possible approximation? $\mathcal{A}_i \neq \mathcal{A}_j$, w not concave?

[Svir17] M. Sviridenko, J. Vondrák, J. Ward, "Optimal Approximation for Submodular and Supermodular Optimization with Bounded Curvature", MOR 17

Main result

Main result

Game theory can be used to produce algorithms that are:

Main result

Game theory can be used to produce algorithms that are:
distributed

Main result

Game theory can be used to produce algorithms that are: distributed, efficient

Main result

Game theory can be used to produce algorithms that are:
distributed, efficient, match/improve existing approximations

Main result

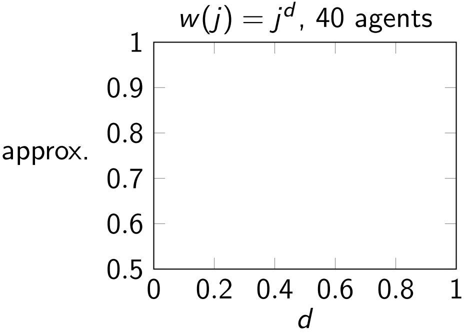
Game theory can be used to produce algorithms that are: distributed, efficient, match/improve existing approximations

Example: - $\#$ agents ≤ 40
- $w(j) = j^d$, d varies in $[0, 1]$

Main result

Game theory can be used to produce algorithms that are:
distributed, efficient, match/improve existing approximations

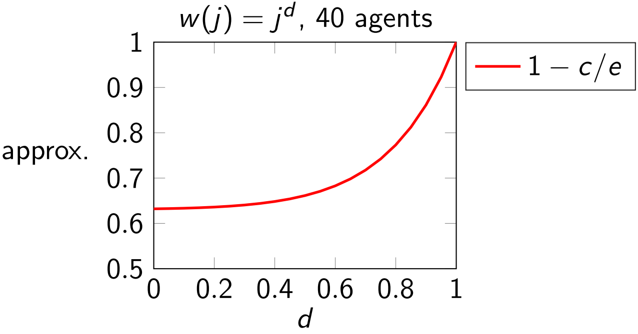
Example: - # agents ≤ 40
- $w(j) = j^d$, d varies in $[0, 1]$



Main result

Game theory can be used to produce algorithms that are: distributed, efficient, match/improve existing approximations

Example: - # agents ≤ 40
- $w(j) = j^d$, d varies in $[0, 1]$

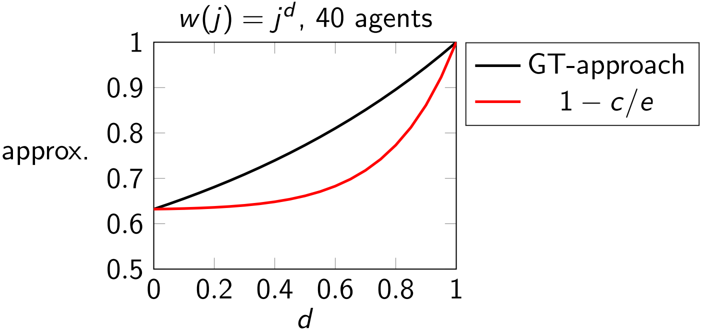


Main result

Game theory can be used to produce algorithms that are: distributed, efficient, match/improve existing approximations

Example: - # agents ≤ 40

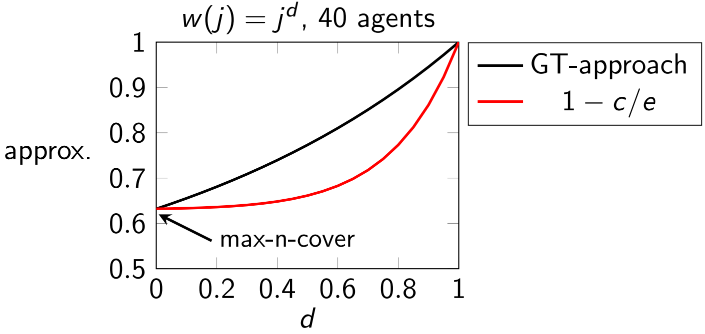
- $w(j) = j^d$, d varies in $[0, 1]$



Main result

Game theory can be used to produce algorithms that are: distributed, efficient, match/improve existing approximations

Example: - # agents ≤ 40
- $w(j) = j^d$, d varies in $[0, 1]$

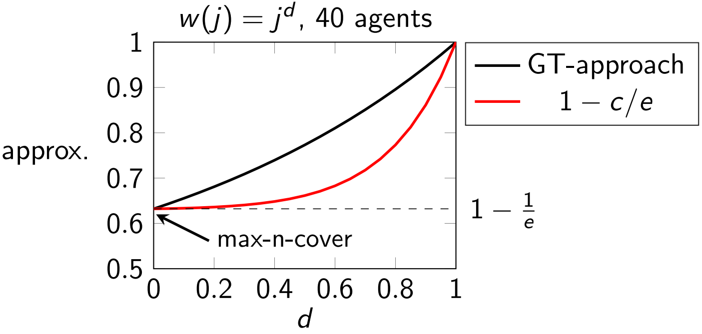


Main result

Game theory can be used to produce algorithms that are: distributed, efficient, match/improve existing approximations

Example: - # agents ≤ 40

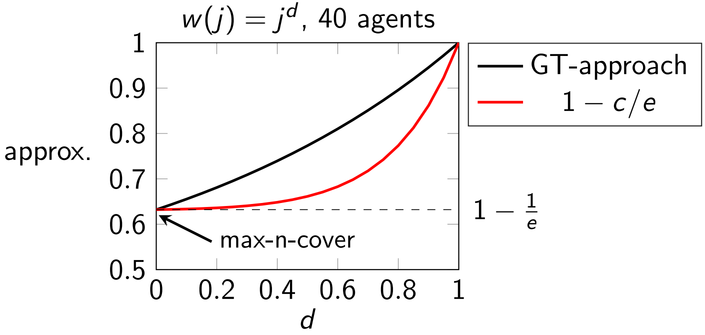
- $w(j) = j^d$, d varies in $[0, 1]$



Main result

Game theory can be used to produce algorithms that are:
distributed, efficient, match/improve existing approximations

Example: - $\#$ agents ≤ 40
- $w(j) = j^d$, d varies in $[0, 1]$



[Pac18a] DP, R.Chandan, J. Marden "Distributed resource allocation through utility design -Part I: optimizing the performance certificates via the price of anarchy", ArXiv 2018

[Pac18b] DP, J. Marden "- Part II: applications to submodular, supermodular and set covering problems", ArXiv 2018

Outline

The game-theoretic approach

The game-theoretic approach

$$\max W(a)$$

- ▷ Distributed algorithm
- ▷ Good approximation
- ▷ Polytime

The game-theoretic approach

Game design

$$\max W(a)$$

- Distributed algorithm
- Good approximation
- Polytime

The game-theoretic approach

Game design

$$\max W(a)$$

- Distributed algorithm
- Good approximation
- Polytime

The game-theoretic approach

$$\max W(a)$$

- ▷ Distributed algorithm
- ▷ Good approximation
- ▷ Polytime

Game design

Design a game (agents, constraints, utilities)

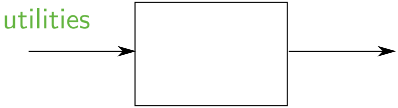
The game-theoretic approach

Game design

Design a game
(agents, constraints, utilities)

$$\max W(a)$$

- Distributed algorithm
- Good approximation
- Polytime



The game-theoretic approach

$$\max W(a)$$

- Distributed algorithm
- Good approximation
- Polytime

Game design

Design a game
(agents, constraints, utilities)

utilities

game

The game-theoretic approach

$$\max W(a)$$

- Distributed algorithm
- Good approximation
- Polytime

Game design

Design a game
(agents, constraints, utilities)

$$\xrightarrow{\text{utilities}} \boxed{\text{game}} \xrightarrow{\text{equilibria}}$$

The game-theoretic approach

$$\max W(a)$$

- Distributed algorithm
- Good approximation
- Polytime

Game design

Design a game
(agents, constraints, utilities)

Requirement:
equilibria have high welfare

$$\text{utilities} \longrightarrow \boxed{\text{game}} \longrightarrow \text{equilibria}$$

The game-theoretic approach

$$\max W(a)$$

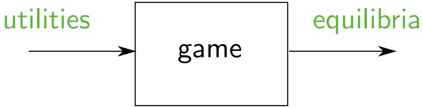
- Distributed algorithm
- Good approximation
- Polytime

Game design

Design a game
(agents, constraints, utilities)

Requirement:
equilibria have high welfare

Use existing algorithms to find an equilibrium



The game-theoretic approach

$$\max W(a)$$

- Distributed algorithm
- Good approximation
- Polytime

Game design

Design a game
(agents, constraints, utilities)

Requirement:
equilibria have high welfare

Use existing algorithms to find an equilibrium

$$\xrightarrow{\text{utilities}} \boxed{\text{game}} \xrightarrow{\text{equilibria}}$$

Utility design and approximation ratio

$u_i(a_i, a_{-i})$

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

How to design f?

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

How to design f? Maximize worst-case performance

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

How to design f? Maximize worst-case performance

Given instance I, fix f

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

How to design f? Maximize worst-case performance

Given instance I, fix $f \to$ game $\mathrm{G}_f = \{\mathrm{I}, f\}$

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

How to design f? Maximize worst-case performance

Given instance I, fix $f \to$ game $\mathrm{G}_f = \{\mathrm{I}, f\} \to \mathrm{NE}(\mathrm{G}_f)$

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

How to design f? Maximize worst-case performance

Given instance I, fix $f \to$ game $\mathrm{G}_f = \{\mathrm{I}, f\} \to \mathrm{NE}(\mathrm{G}_f)$

$$\mathrm{PoA}(f) = \qquad \frac{\min_{a \in \mathrm{NE}(\mathrm{G}_f)} W(a)}{W(a_{\mathrm{opt}})} \leq 1$$

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

How to design f? Maximize worst-case performance

Given instance I, fix $f \to$ game $\mathrm{G}_f = \{\mathrm{I}, f\} \to \mathrm{NE}(\mathrm{G}_f)$

$$\mathrm{PoA}(f) = \inf_{\mathrm{G}_f \,:\, \#\text{agents} \leq n} \frac{\min_{a \in \mathrm{NE}(\mathrm{G}_f)} W(a)}{W(a_{\mathrm{opt}})} \leq 1$$

Utility design and approximation ratio

$$u_i(a_i, a_{-i}) = \sum_{r \in a_i} v_r w(|a|_r) f(|a|_r) \qquad f : \mathbb{N} \to \mathbb{R}_{\geq 0} \quad (\textit{distributed})$$

How to design f? Maximize worst-case performance

Given instance I, fix $f \to$ game $\mathrm{G}_f = \{\mathrm{I}, f\} \to \mathrm{NE}(\mathrm{G}_f)$

$$\mathrm{PoA}(f) = \inf_{\mathrm{G}_f \,:\, \#\text{agents} \leq n} \frac{\min_{a \in \mathrm{NE}(\mathrm{G}_f)} W(a)}{W(a_{\mathrm{opt}})} \leq 1$$

$\boxed{\mathrm{PoA}(f) \text{ is the approx. ratio of any equilibrium-computing algorithm}}$

Utility design reduces to

Given f, characterize or bound $\mathrm{PoA}(f)$

Utility design reduces to

Utility design reduces to

- Given f, characterize or bound $\mathrm{PoA}(f)$
- Find f with highest $\mathrm{PoA}(f)$

$$\text{Utility design reduces to} \begin{cases} \boxed{\text{Given } f\text{, characterize or bound } \mathrm{PoA}(f)} \\ \text{Find } f \text{ with highest } \mathrm{PoA}(f) \end{cases}$$

Characterizing the price of anarchy

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory
- difficult to compute

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory
- difficult to compute
- bounds are available in special cases

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory
- difficult to compute
- bounds are available in special cases
- bounds do not explicitly depend on f

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory
- difficult to compute
- bounds are available in special cases
- bounds do not explicitly depend on f
- smoothness not applicable

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory
- difficult to compute
- bounds are available in special cases
- bounds do not explicitly depend on f
- smoothness not applicable
- tightness?

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory
- difficult to compute
- bounds are available in special cases
- bounds do not explicitly depend on f
- smoothness not applicable
- tightness?

Theorem (Characterization of $\mathrm{PoA}(f)$) [Pac18a],[Pac18b]

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory
- difficult to compute
- bounds are available in special cases
- bounds do not explicitly depend on f
- smoothness not applicable
- tightness?

Theorem (Characterization of $\mathrm{PoA}(f)$) [Pac18a],[Pac18b]

$\mathrm{PoA}(f)$ *is the solution to a tractable LP in 2 variables,* $\mathcal{O}(n^2)$ *constraints*

Characterizing the price of anarchy

The quantity we wish to compute: $\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(a_{opt})} \right)$

- well studied in game theory
- difficult to compute
- bounds are available in special cases
- bounds do not explicitly depend on f
- smoothness not applicable
- tightness?

Theorem (Characterization of $\mathrm{PoA}(f)$) [Pac18a],[Pac18b]

$\mathrm{PoA}(f)$ *is the solution to a tractable LP in 2 variables,* $\mathcal{O}(n^2)$ *constraints*

▷ *LP involves all the components* $w(j)$ *and* $f(j)$

Proof Sketch - Part 1/4

$\boxed{\textbf{Idea}\text{: transform the definition of } \mathrm{PoA} \text{ itself into a LP}}$

Proof Sketch - Part 1/4

$\boxed{\textbf{Idea}\text{: transform the definition of } \mathrm{PoA} \text{ itself into a LP}}$

Four steps towards the goal:

Proof Sketch - Part 1/4

Idea: transform the definition of PoA itself into a LP

Four steps towards the goal:

1. $\mathrm{PoA}(f)$ is the same of the price of anarchy over a reduced class of games where each agent has **only two feasible allocations**

Proof Sketch - Part 1/4

Idea: transform the definition of PoA itself into a LP

Four steps towards the goal:

1. $\mathrm{PoA}(f)$ is the same of the price of anarchy over a reduced class of games where each agent has **only two feasible allocations**
   i.e. we can reduce to $\tilde{\mathcal{A}}_i = \{e_i, o_i\}$ with e_i the worst NE

Proof Sketch - Part 1/4

Idea: transform the definition of PoA itself into a LP

Four steps towards the goal:

1. $\mathrm{PoA}(f)$ is the same of the price of anarchy over a reduced class of games where each agent has **only two feasible allocations**
i.e. we can reduce to $\tilde{\mathcal{A}}_i = \{e_i, o_i\}$ with e_i the worst NE

$$\mathrm{PoA}(f) = \inf_{G \in \mathcal{G}} \left(\frac{\min_{a \in NE(G)} W(a)}{\max_{a \in \mathcal{A}} W(a)} \right)$$

Proof Sketch - Part 1/4

Idea: transform the definition of PoA itself into a LP

Four steps towards the goal:

1. $\mathrm{PoA}(f)$ is the same of the price of anarchy over a reduced class of games where each agent has **only two feasible allocations**
   i.e. we can reduce to $\tilde{\mathcal{A}}_i = \{e_i, o_i\}$ with e_i the worst NE

$$\mathrm{PoA}(f) = \inf_{G \in \tilde{\mathcal{G}}} \left(\frac{\min_{a \in NE(G)} W(a)}{W(o)} \right)$$

Proof Sketch - Part 1/4

Idea: transform the definition of PoA itself into a LP

Four steps towards the goal:

1. $\mathrm{PoA}(f)$ is the same of the price of anarchy over a reduced class of games where each agent has **only two feasible allocations**
   i.e. we can reduce to $\tilde{\mathcal{A}}_i = \{e_i, o_i\}$ with e_i the worst NE

$$\mathrm{PoA}(f) = \inf_{G \in \tilde{\mathcal{G}}} \left(\frac{W(e)}{W(o)} \right)$$

Proof Sketch - Part 1/4

Idea: transform the definition of PoA itself into a LP

Four steps towards the goal:

1. $\mathrm{PoA}(f)$ is the same of the price of anarchy over a reduced class of games where each agent has **only two feasible allocations** i.e. we can reduce to $\tilde{\mathcal{A}}_i = \{e_i, o_i\}$ with e_i the worst NE

$$\mathrm{PoA}(f) = \inf_{G \in \tilde{\mathcal{G}}} \left(\frac{W(e)}{W(o)} \right) \quad \text{s.t.} \quad u_i(e) \geq u_i(o_i, e_{-i}) \quad \forall i$$

Proof Sketch - Part 1/4

Idea: transform the definition of PoA itself into a LP

Four steps towards the goal:

1. $\mathrm{PoA}(f)$ is the same of the price of anarchy over a reduced class of games where each agent has **only two feasible allocations** i.e. we can reduce to $\tilde{\mathcal{A}}_i = \{e_i, o_i\}$ with e_i the worst NE

$$\mathrm{PoA}(f) = \inf_{G \in \tilde{\mathcal{G}}} \left(\frac{W(e)}{W(o)} \right) \quad \text{s.t.} \quad u_i(e) \geq u_i(o_i, e_{-i}) \quad \forall i$$

2. Relax the previous program

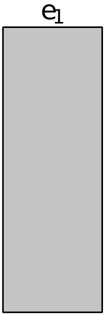
$$\mathrm{PoA}(f) = \inf_{G \in \tilde{\mathcal{G}}} \frac{W(e)}{W(o)}$$
$$\text{s.t.} \quad \sum_i u_i(e) \geq \sum_i u_i(o_i, e_{-i})$$

Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$

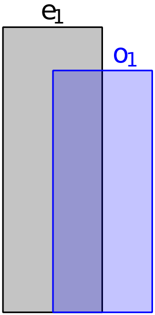
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



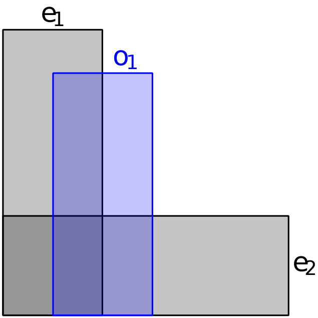
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



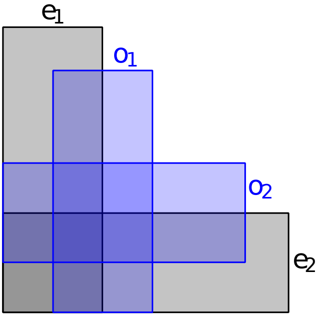
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



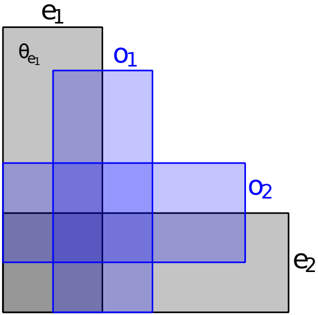
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



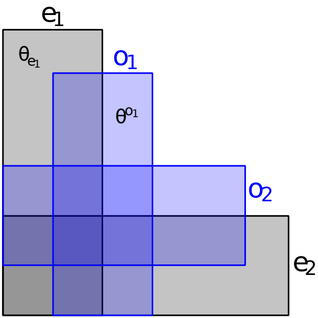
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



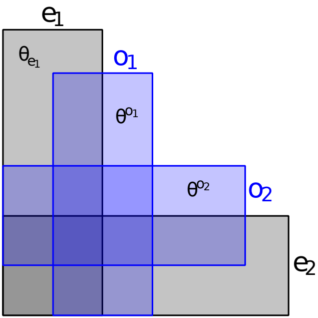
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



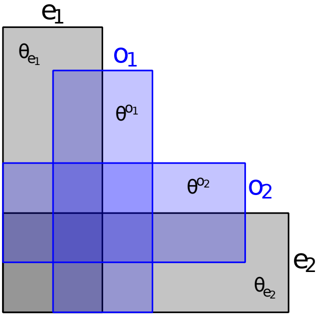
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



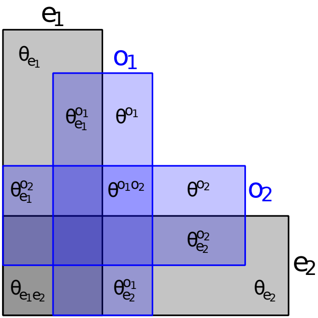
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



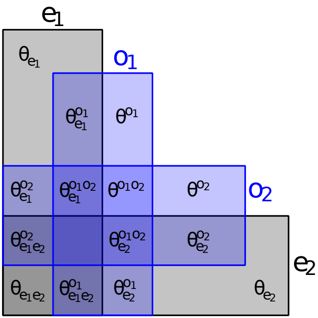
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



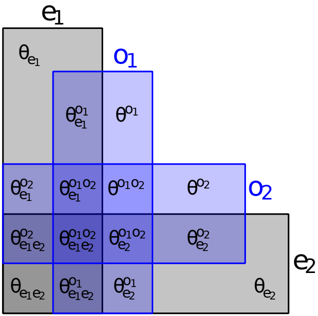
Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



Proof Sketch - Part 2/4

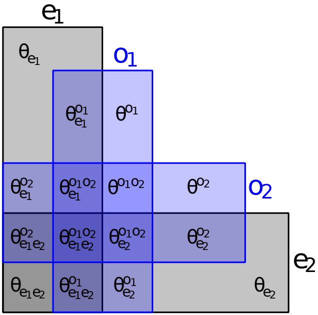
3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



Proof Sketch - Part 2/4

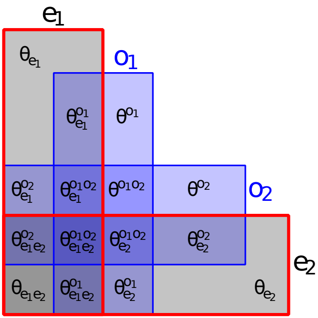
3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$

Variables θ allow to compute $W(a)$, $u_i(a)$ in **all allocations**



Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$

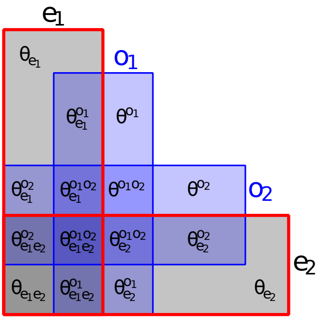


Variables θ allow to compute $W(a)$, $u_i(a)$ in **all allocations**, e.g.

$$
\begin{aligned}
&W(e_1, e_2) \\
&= (\theta_{e_1} + \theta_{e_1}^{o_1} + \theta_{e_1}^{o_2} + \theta_{e_1}^{o_1 o_2})w(1) \\
&+ (\theta_{e_2} + \theta_{e_2}^{o_2} + \theta_{e_2}^{o_1} + \theta_{e_2}^{o_1 o_2})w(1) \\
&+ (\theta_{e_1 e_2} + \theta_{e_1 e_2}^{o_1} + \theta_{e_1 e_2}^{o_2} + \theta_{e_1 e_2}^{o_1 o_2})w(2)
\end{aligned}
$$

Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



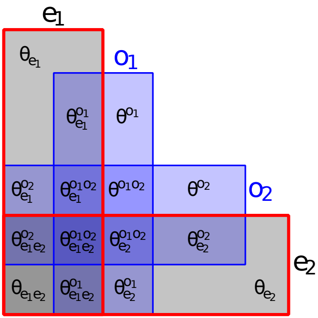
Variables θ allow to compute $W(a)$, $u_i(a)$ in **all allocations**, e.g.

$$W(e_1, e_2) = (\theta_{e_1} + \theta_{e_1}^{o_1} + \theta_{e_1}^{o_2} + \theta_{e_1}^{o_1 o_2})w(1) + (\theta_{e_2} + \theta_{e_2}^{o_2} + \theta_{e_2}^{o_1} + \theta_{e_2}^{o_1 o_2})w(1) + (\theta_{e_1 e_2} + \theta_{e_1 e_2}^{o_1} + \theta_{e_1 e_2}^{o_2} + \theta_{e_1 e_2}^{o_1 o_2})w(2)$$

$$u_1(e_1, e_2) = (\theta_{e_1} + \theta_{e_1}^{o_1} + \theta_{e_1}^{o_2} + \theta_{e_1}^{o_1 o_2})w(1)f(1) + (\theta_{e_1 e_2} + \theta_{e_1 e_2}^{o_1} + \theta_{e_1 e_2}^{o_1 o_2} + \theta_{e_1 e_2}^{o_2})w(2)f(2)$$

Proof Sketch - Part 2/4

3. How to describe an instance? Need to describe $W(a)$, $u_i(a)$ on $\tilde{\mathcal{A}}$



Variables θ allow to compute $W(a)$, $u_i(a)$ in **all allocations**, e.g.

$$W(e_1, e_2)$$
$$= (\theta_{e_1} + \theta_{e_1}^{o_1} + \theta_{e_1}^{o_2} + \theta_{e_1}^{o_1 o_2})w(1)$$
$$+ (\theta_{e_2} + \theta_{e_2}^{o_2} + \theta_{e_2}^{o_1} + \theta_{e_2}^{o_1 o_2})w(1)$$
$$+ (\theta_{e_1 e_2} + \theta_{e_1 e_2}^{o_1} + \theta_{e_1 e_2}^{o_2} + \theta_{e_1 e_2}^{o_1 o_2})w(2)$$

$$u_1(e_1, e_2)$$
$$= (\theta_{e_1} + \theta_{e_1}^{o_1} + \theta_{e_1}^{o_2} + \theta_{e_1}^{o_1 o_2})w(1)f(1)$$
$$+ (\theta_{e_1 e_2} + \theta_{e_1 e_2}^{o_1} + \theta_{e_1 e_2}^{o_1 o_2} + \theta_{e_1 e_2}^{o_2})w(2)f(2)$$

Issue: #weights is exponential!

Proof Sketch - Part 3/4

4. use **reduced variables** for $\mathbf{W}(\mathbf{e})$, $\mathbf{W}(\mathbf{o})$, $\sum_{\mathbf{i}} \mathbf{u}_{\mathbf{i}}(\mathbf{e}) - \mathbf{u}_{\mathbf{i}}(\mathbf{o}_{\mathbf{i}}, \mathbf{e}_{-\mathbf{i}})$
   $\rightarrow$ define $\theta(a, x, b) \in \mathbb{R}_{\geq 0}$ for $1 \leq a + x + b \leq n, a, x, b \in \{1, \ldots, n\}$

Proof Sketch - Part 3/4

4. use **reduced variables** for $\mathbf{W(e)}$, $\mathbf{W(o)}$, $\sum_{\mathbf{i}} \mathbf{u_i(e) - u_i(o_i, e_{-i})}$
   $\rightarrow$ define $\theta(a, x, b) \in \mathbb{R}_{\geq 0}$ for $1 \leq a + x + b \leq n$, $a, x, b \in \{1, \ldots, n\}$

$$W(e) = \sum_{a,x,b} \mathbb{1}_{\{a+x \geq 1\}} w(a + x)\theta(a, x, b)$$

Proof Sketch - Part 3/4

4. use **reduced variables** for $\mathbf{W(e)}$, $\mathbf{W(o)}$, $\sum_{\mathbf{i}} \mathbf{u_i(e) - u_i(o_i, e_{-i})}$
   $\rightarrow$ define $\theta(a, x, b) \in \mathbb{R}_{\geq 0}$ for $1 \leq a + x + b \leq n$, $a, x, b \in \{1, \ldots, n\}$

$$W(e) = \sum_{a,x,b} \mathbb{1}_{\{a+x\geq 1\}} w(a+x)\theta(a,x,b)$$

$$W(o) = \sum_{a,x,b} \mathbb{1}_{\{b+x\geq 1\}} w(b+x)\theta(a,x,b)$$

Proof Sketch - Part 3/4

4. use **reduced variables** for $\mathbf{W}(\mathbf{e})$, $\mathbf{W}(\mathbf{o})$, $\sum_{\mathbf{i}} \mathbf{u}_{\mathbf{i}}(\mathbf{e}) - \mathbf{u}_{\mathbf{i}}(\mathbf{o}_{\mathbf{i}}, \mathbf{e}_{-\mathbf{i}})$
   $\to$ define $\theta(a, x, b) \in \mathbb{R}_{\geq 0}$ for $1 \leq a + x + b \leq n, a, x, b \in \{1, \ldots, n\}$

$$W(e) = \sum_{a,x,b} \mathbb{1}_{\{a+x \geq 1\}} w(a+x)\theta(a, x, b)$$

$$W(o) = \sum_{a,x,b} \mathbb{1}_{\{b+x \geq 1\}} w(b+x)\theta(a, x, b)$$

$$\text{equil.} = \sum_{a,x,b} [af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)]\theta(a, x, b)$$

Proof Sketch - Part 3/4

4. use **reduced variables** for $\mathbf{W}(\mathbf{e})$, $\mathbf{W}(\mathbf{o})$, $\sum_{\mathbf{i}} \mathbf{u_i}(\mathbf{e}) - \mathbf{u_i}(\mathbf{o_i}, \mathbf{e_{-i}})$
   $\rightarrow$ define $\theta(a, x, b) \in \mathbb{R}_{\geq 0}$ for $1 \leq a + x + b \leq n$, $a, x, b \in \{1, \ldots, n\}$

$$W(e) = \sum_{a,x,b} \mathbb{1}_{\{a+x\geq 1\}} w(a+x)\theta(a, x, b)$$

$$W(o) = \sum_{a,x,b} \mathbb{1}_{\{b+x\geq 1\}} w(b+x)\theta(a, x, b)$$

$$\text{equil.} = \sum_{a,x,b} [af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)]\theta(a, x, b)$$

The program becomes

Proof Sketch - Part 3/4

4. use **reduced variables** for $\mathbf{W(e)}$, $\mathbf{W(o)}$, $\sum_{\mathbf{i}} \mathbf{u_i(e) - u_i(o_i, e_{-i})}$
   $\rightarrow$ define $\theta(a, x, b) \in \mathbb{R}_{\geq 0}$ for $1 \leq a + x + b \leq n$, $a, x, b \in \{1, \ldots, n\}$

$$W(e) = \sum_{a,x,b} \mathbb{1}_{\{a+x\geq 1\}} w(a+x)\theta(a,x,b)$$

$$W(o) = \sum_{a,x,b} \mathbb{1}_{\{b+x\geq 1\}} w(b+x)\theta(a,x,b)$$

$$\text{equil.} = \sum_{a,x,b} [af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)]\theta(a,x,b)$$

The program becomes $\mathrm{PoA}(f) = \inf_{\theta(a,x,b)\geq 0} \dfrac{W(e)}{W(o)}$

$$\text{s.t.} \quad \sum_i u_i(e) - u_i(o_i, e_{-i}) \geq 0$$

Proof Sketch - Part 3/4

4. use **reduced variables** for $\mathbf{W(e)}$, $\mathbf{W(o)}$, $\sum_{\mathbf{i}} \mathbf{u_i(e) - u_i(o_i, e_{-i})}$
   $\rightarrow$ define $\theta(a, x, b) \in \mathbb{R}_{\geq 0}$ for $1 \leq a + x + b \leq n$, $a, x, b \in \{1, \ldots, n\}$

$$W(e) = \sum_{a,x,b} \mathbb{1}_{\{a+x\geq 1\}} w(a+x)\theta(a,x,b)$$

$$W(o) = \sum_{a,x,b} \mathbb{1}_{\{b+x\geq 1\}} w(b+x)\theta(a,x,b)$$

$$\text{equil.} = \sum_{a,x,b} [af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)]\theta(a,x,b)$$

The program becomes $\mathrm{PoA}(f) = \inf_{\theta(a,x,b)\geq 0} \dfrac{1}{W(o)}$

$$\text{s.t.} \quad \sum_i u_i(e) - u_i(o_i, e_{-i}) \geq 0$$

$$W(e) = 1$$

Proof Sketch - Part 4/4: Primal LP

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$W^\star = \sup_{\theta(a,x,b)} \sum_{a,x,b} \mathbb{1}_{\{b+x\geq 1\}} w(b+x)\theta(a,x,b)$$

$$\text{s.t.} \sum_{a,x,b} [af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)]\theta(a,x,b) \geq 0$$

$$\sum_{a,x,b} \mathbb{1}_{\{a+x\geq 1\}} w(a+x)\theta(a,x,b) = 1$$

$$\theta(a,x,b) \geq 0 \quad \forall (a,x,b) \in \mathcal{I}\,.$$

Dual LP

Dual LP

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$W^\star = \inf_{\lambda \in \mathbb{R}_{\geq 0},\, \mu \in \mathbb{R}} \ \mu$$

$$\begin{aligned}\text{s.t.}\quad & \mathbb{1}_{\{b+x\geq 1\}} w(b+x) - \mu \mathbb{1}_{\{a+x \geq 1\}} w(a+x) + \\ & + \lambda[af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)] \leq 0 \\ & \qquad\qquad\qquad\qquad\qquad\qquad \forall (a, x, b) \in \mathcal{I}\end{aligned}$$

Dual LP

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$\begin{aligned} W^\star = \inf_{\lambda \in \mathbb{R}_{\geq 0},\, \mu \in \mathbb{R}} \quad & \mu \\ \text{s.t.} \quad & \mathbb{1}_{\{b+x\geq 1\}} w(b+x) - \mu \mathbb{1}_{\{a+x \geq 1\}} w(a+x) + \\ & +\lambda[af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)] \leq 0 \\ & \qquad \forall (a,x,b) \in \partial\mathcal{I} \end{aligned}$$

Dual LP

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$W^\star = \inf_{\lambda \in \mathbb{R}_{\geq 0},\, \mu \in \mathbb{R}} \; \mu$$

$$\text{s.t.} \quad \mathbb{1}_{\{b+x\geq 1\}} w(b+x) - \mu \mathbb{1}_{\{a+x\geq 1\}} w(a+x) +$$
$$+\lambda[af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)] \leq 0$$
$$\forall (a, x, b) \in \partial\mathcal{I}$$

- ▷ 2 decision variables, $\mathcal{O}(n^2)$ constraints

Dual LP

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$W^\star = \inf_{\lambda \in \mathbb{R}_{\geq 0},\, \mu \in \mathbb{R}} \ \mu$$

$$\text{s.t.} \quad \mathbb{1}_{\{b+x\geq 1\}} w(b+x) - \mu \mathbb{1}_{\{a+x\geq 1\}} w(a+x) +$$

$$+\lambda[af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)] \leq 0$$

$$\forall (a,x,b) \in \partial\mathcal{I}$$

- 2 decision variables, $\mathcal{O}(n^2)$ constraints
- observe the special structure i.e. $\min_{\lambda,\mu} \mu$ subject to $\mu \geq \dots$

Dual LP

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$W^\star = \inf_{\lambda \in \mathbb{R}_{\geq 0},\, \mu \in \mathbb{R}} \ \mu$$

$$\text{s.t.} \ \ \mathbb{1}_{\{b+x\geq 1\}} w(b+x) - \mu \mathbb{1}_{\{a+x\geq 1\}} w(a+x) +$$
$$+\lambda[af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)] \leq 0$$
$$\forall (a,x,b) \in \partial\mathcal{I}$$

- 2 decision variables, $\mathcal{O}(n^2)$ constraints
- observe the special structure i.e. $\min_{\lambda,\mu} \mu$ subject to $\mu \geq \dots$
- gives PoA for e.g., $f_{\mathrm{sv}}(j) = 1/j$, $f_{\mathrm{mc}}(j) = 1 - w(j-1)/w(j)$

PoA: connection with existing literature

Covering Games: Approximation through Non-Cooperation ⋆

Martin Gairing

Department of Computer Science, University of Liverpool, U.K.
m.gairing@liverpool.ac.uk

Abstract. We propose approximation algorithms under game-theoretic considerations. We indroduce and study the *general covering problem* which is a natural generalization of the well-studied *max-n-cover problem*. In the general covering problem, we are given a universal set of weighted elements E and n collections of subsets of the elements. The task is to choose one subset from each collection such that the total weight of their union is as large as possible. In our game-theoretic setting, the choice in each collection is made by an independent player. For covering an element, the players receive a payoff defined by a non-increasing *utility sharing function*. This function defines the fraction that each covering player receives from the weight of the elements.
We show how to construct a utility sharing function such that every Nash Equilibrium approximates the optimal solution by a factor of $1 - \frac{1}{e}$. We also prove that any sequence of unilateral improving steps is polynomially bounded. This gives rise to a polynomial-time local search approximation algorithm whose approximation ratio is best possible.

PoA: connection with existing literature

Covering Games: Approximation through Non-Cooperation ⋆

Martin Gairing

IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 59, NO. 3, MARCH 2014 571

Generalized Efficiency Bounds in Distributed Resource Allocation

Jason R. Marden and Tim Roughgarden

***Abstract*—Game theory is emerging as a popular tool for distributed control of multiagent systems. To take advantage of these game theoretic tools, the interactions of the autonomous agents must be designed within a game-theoretic environment. A central component of this game-theoretic design is the assignment of a local utility function to each agent. One promising approach to utility design is assigning each agent a utility function according to the agent's *Shapley value*. This method frequently results in games that possess many desirable features, such as the existence of pure Nash equilibria with near-optimal efficiency. In this paper, we explore the relationship between the Shapley value utility design and the resulting efficiency of both pure Nash equilibria and coarse correlated equilibria. To study this relationship, we introduce a simple class of resource allocation problems. Within this class, we derive an explicit relationship between the structure of the resource allocation problem and the efficiency of the resulting equilibria. Lastly, we derive a bicriteria bound for this class of resource allocation problems—a bound on the value of the optimal allocation relative to the value of an equilibrium allocation with additional agents.**

in large-scale engineering systems, where a centralized control approach is undesirable or even infeasible. For example, a centralized control approach may be impossible for the aforementioned sensor allocation problem because of the complexity associated with a potentially large number of sensors, the vastness/uncertainty of the mission space, or potential stealth requirements that restrict communication capabilities. A more desirable control approach is to establish a distributed control algorithm that allows the sensors to allocate themselves effectively over the mission space without the need for global intervention [14], [15]. Such an algorithm would eliminate the need for centralized communication and introduce an inherent robustness to communication failures, sensor failures, and environmental uncertainties. While desirable, establishing such a distributed control algorithm comes with its share of challenges. Is it possible to characterize the global behavior that

PoA: connection with existing literature

Covering Games: Approximation through Non-Cooperation *

Martin Gairing

IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 59, NO. 3, MARCH 2014 571

Generalized Efficiency Bounds in Distributed Resource Allocation

Jason R. Marden and Tim Roughgarden

Optimal Approximation for Submodular and Supermodular Optimization with Bounded Curvature

Maxim Sviridenko,[a] Jan Vondrák,[b] Justin Ward[c]

[a] Yahoo! Labs, New York, New York 10018; [b] Stanford University, Stanford, California 94305; [c] Ecole Polytechnique Federale de Lausanne, 1015 Lausanne, Switzerland

Contact: sviri@yahoo-inc.com (MS); jvondrak@stanford.edu (JV); justin.ward@epfl.ch (JW)

Received: December 30, 2014
Revised: June 20, 2016
Accepted: September 8, 2016
Published Online in Articles in Advance: May 16, 2017

MSC2010 Subject Classification: Primary: 90C27; secondary: 68W25
OR/MS Subject Classification: Primary: analysis of algorithms; secondary: mathematics: functions

https://doi.org/10.1287/moor.2016.0842

Abstract. We design new approximation algorithms for the problems of optimizing submodular and supermodular functions subject to a single matroid constraint. Specifically, we consider the case in which we wish to maximize a monotone increasing submodular function or minimize a monotone decreasing supermodular function with a bounded total curvature c. Intuitively, the parameter c represents how nonlinear a function f is: when $c = 0$, f is linear, while for $c = 1$, f may be an arbitrary monotone increasing submodular function. For the case of submodular maximization with total curvature c, we obtain a $(1 - c/e)$-approximation—the first improvement over the greedy algorithm of of Conforti and Cornuéjols from 1984, which holds for a cardinality constraint, as well as a recent analogous result for an arbitrary matroid constraint.

Our approach is based on modifications of the continuous greedy algorithm and nonoblivious local search, and allows us to approximately maximize the sum of a nonnegative, monotone increasing submodular function and a (possibly negative) linear function. We show how to reduce both submodular maximization and supermodular minimization to this general problem when the objective function has bounded total curvature. We prove that the approximation results we obtain are the best possible in the value oracle model, even in the case of a cardinality constraint.

We define an extension of the notion of curvature to general monotone set functions and show a $(1 - c)$-approximation for maximization and a $1/(1 - c)$-approximation for minimization cases. Finally, we give two concrete applications of our results in the settings of maximum entropy sampling, and the column-subset selection problem.

Utility design reduces to

Given f, characterize or bound $\mathrm{PoA}(f)$

Find f with highest $\mathrm{PoA}(f)$

Utility design reduces to

Given f, characterize or bound $\mathrm{PoA}(f)$

$\boxed{\text{Find } f \text{ with highest } \mathrm{PoA}(f)}$

Optimal price of anarchy

Optimal price of anarchy

Corollary (Optimizing PoA) [Pac18a], [Pac18b]

Determining $f \in \mathbb{R}^n_{\geq 0}$ maximizing $\mathrm{PoA}(f)$ is a tractable linear program

Optimal price of anarchy

Corollary (Optimizing PoA) [Pac18a], [Pac18b]

Determining $f \in \mathbb{R}^n_{\geq 0}$ maximizing $\mathrm{PoA}(f)$ is a tractable linear program

Proof.

Optimal price of anarchy

Corollary (Optimizing PoA) [Pac18a], [Pac18b]

Determining $f \in \mathbb{R}^n_{\geq 0}$ maximizing $\mathrm{PoA}(f)$ is a tractable linear program

Proof.

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$\begin{aligned} W^\star = \quad & \inf_{\lambda \in \mathbb{R}_{\geq 0},\, \mu \in \mathbb{R}} \ \mu \\ \text{s.t.} \quad & \mathbb{1}_{\{b+x\geq 1\}} w(b+x) - \mu \mathbb{1}_{\{a+x\geq 1\}} w(a+x) + \\ & + \lambda[af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)] \leq 0 \\ & \qquad\qquad \forall (a,x,b) \in \partial\mathcal{I} \end{aligned}$$

Optimal price of anarchy

Corollary (Optimizing PoA) [Pac18a], [Pac18b]

Determining $f \in \mathbb{R}^n_{\geq 0}$ maximizing $\mathrm{PoA}(f)$ is a tractable linear program

Proof.

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$W^\star = \min_{f \in \mathbb{R}^n_{\geq 0}} \inf_{\lambda \in \mathbb{R}_{\geq 0},\, \mu \in \mathbb{R}} \mu$$

$$\begin{aligned} \text{s.t.}\quad & \mathbb{1}_{\{b+x\geq 1\}} w(b+x) - \mu \mathbb{1}_{\{a+x\geq 1\}} w(a+x) + \\ & + \lambda[af(a+x)w(a+x) - bf(a+x+1)w(a+x+1)] \leq 0 \\ & \qquad\qquad\qquad\qquad \forall (a,x,b) \in \partial\mathcal{I} \end{aligned}$$

Optimal price of anarchy

Corollary (Optimizing PoA) [Pac18a], [Pac18b]

Determining $f \in \mathbb{R}^n_{\geq 0}$ maximizing $\mathrm{PoA}(f)$ is a tractable linear program

Proof.

$$\mathrm{PoA}(f) = \frac{1}{W^\star}$$

$$\begin{aligned}
W^\star = \min_{f \in \mathbb{R}^n_{\geq 0}} \; & \inf_{\lambda \in \mathbb{R}_{\geq 0},\, \mu \in \mathbb{R}} \; \mu \\
\text{s.t.} \;\; & \mathbb{1}_{\{b+x \geq 1\}} w(b+x) - \mu \mathbb{1}_{\{a+x \geq 1\}} w(a+x) + \\
& + \lambda[a f(a+x) w(a+x) - b f(a+x+1) w(a+x+1)] \leq 0 \\
& \qquad \forall (a, x, b) \in \partial \mathcal{I}
\end{aligned}$$

□

Back to the main result

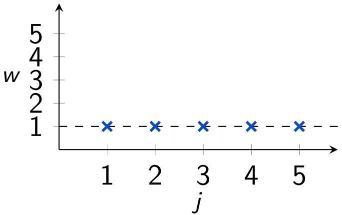
Example:

- $\#$ agents ≤ 40

Back to the main result

Example:

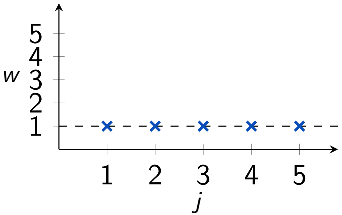
- $\#$ agents ≤ 40
- $w(j) = j^d$, $d = 0$

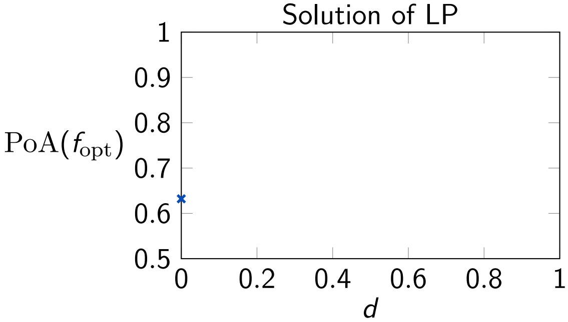


Back to the main result

Example:

- $\#$ agents ≤ 40
- $w(j) = j^d$, $d = 0$

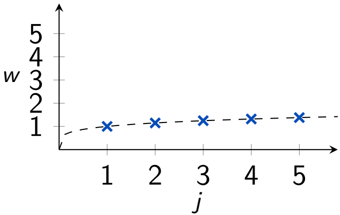


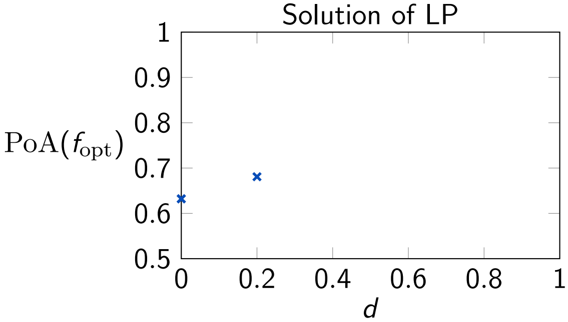


Back to the main result

Example:

- $\#$ agents ≤ 40
- $w(j) = j^d$, $d = 0.2$

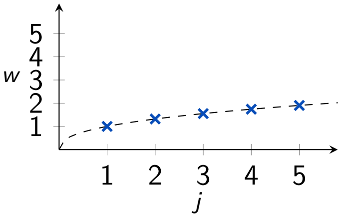


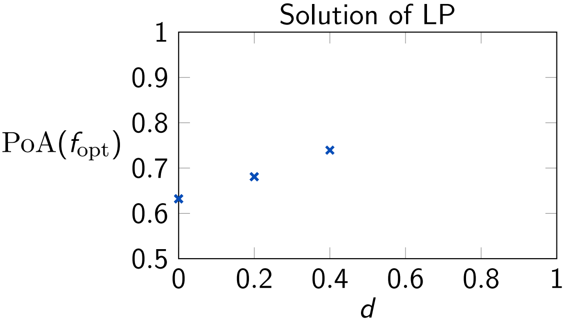


Back to the main result

Example:

- \# agents ≤ 40
- $w(j) = j^d$, $d = 0.4$

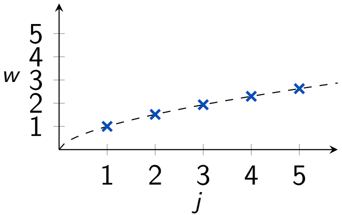


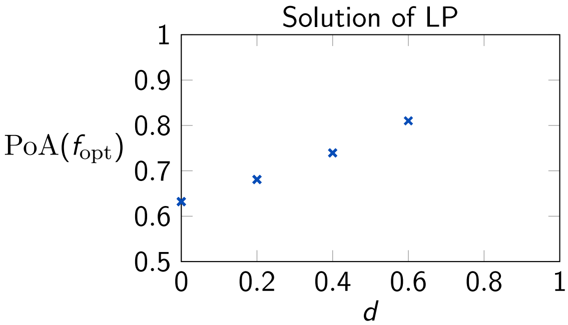


Back to the main result

Example:

- \# agents ≤ 40
- $w(j) = j^d$, $d = 0.6$

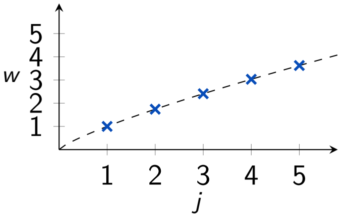


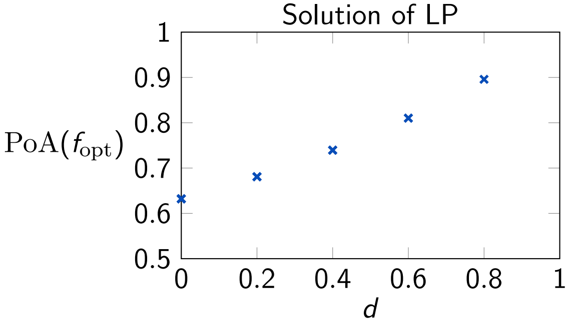


Back to the main result

Example:

- $\#$ agents ≤ 40
- $w(j) = j^d$, $d = 0.8$

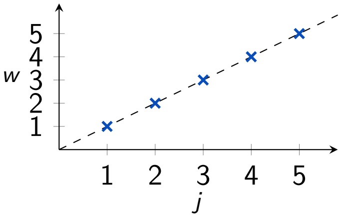


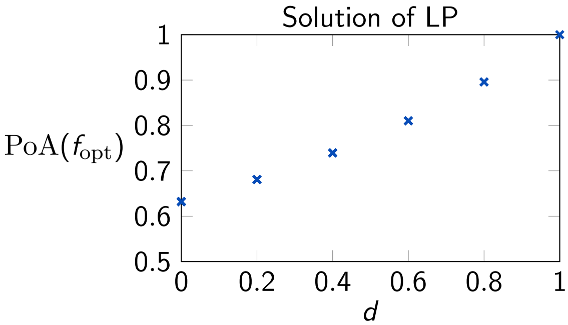


Back to the main result

Example:

- $\#$ agents ≤ 40
- $w(j) = j^d$, $d = 1$

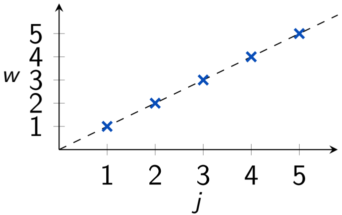


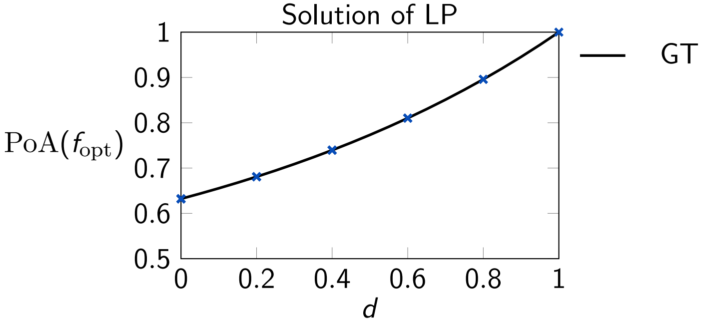


Back to the main result

Example:

- $\#$ agents ≤ 40
- $w(j) = j^d$, $d = 1$

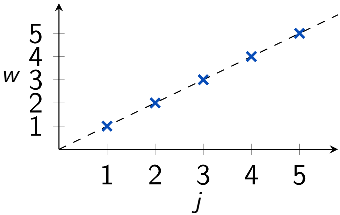


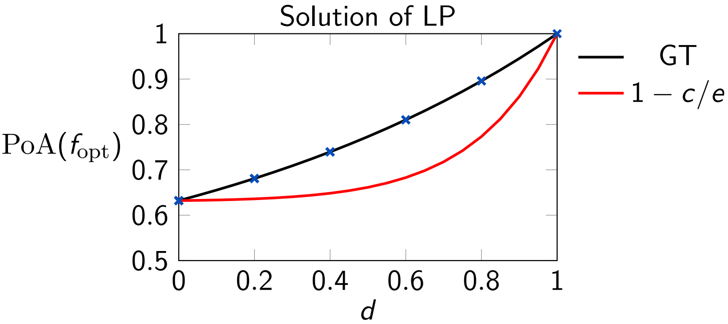


Back to the main result

Example:

- \# agents ≤ 40
- $w(j) = j^d$, $d = 1$





Comparison with other distributions, $\#agents \leq 20$

Comparison with other distributions, $\#agents \leq 20$

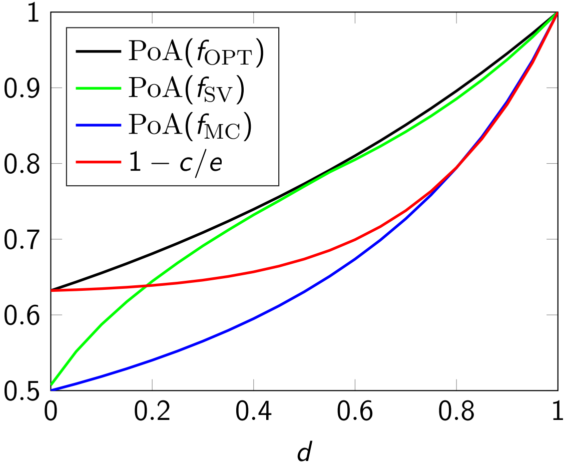
- $f_{\mathrm{SV}}(j) = \frac{1}{j}$

Comparison with other distributions, $\#agents \leq 20$

- $f_{\mathrm{SV}}(j) = \frac{1}{j}$
- $f_{\mathrm{MC}}(j) = 1 - \frac{w(j-1)}{w(j)}$

Comparison with other distributions, $\#agents \leq 20$

- $f_{\mathrm{SV}}(j) = \frac{1}{j}$
- $f_{\mathrm{MC}}(j) = 1 - \frac{w(j-1)}{w(j)}$



Conclusions and Outlook

Conclusions and Outlook

The problem: Generalized Multiagent Maximum Coverage

Conclusions and Outlook

The problem:	Generalized Multiagent Maximum Coverage
The approach:	Approximation through game theory

Conclusions and Outlook

The problem: Generalized Multiagent Maximum Coverage

The approach: Approximation through game theory

- Computing the exact price of anarchy
- Optimizing the price of anarchy

Conclusions and Outlook

The problem: Generalized Multiagent Maximum Coverage

The approach: Approximation through game theory

- Computing the exact price of anarchy
- Optimizing the price of anarchy

The contribution: Distributed algorithms, improved performance

Conclusions and Outlook

The problem: Generalized Multiagent Maximum Coverage

The approach: Approximation through game theory

- Computing the exact price of anarchy
- Optimizing the price of anarchy

The contribution: Distributed algorithms, improved performance

Outlook: Extension to

- Coarse correlated equilibria
- More general W

Thank you

people.ee.ethz.ch/~dariop

Thank you

`people.ee.ethz.ch/~dariop`

[Pac18a] D. Paccagnan, R. Chandan and J.R. Marden. "Distributed resource allocation through utility design - Part I: optimizing the performance certificates via the price of anarchy". *ArXiv*, **2018**.

[Pac18b] D. Paccagnan and J.R. Marden. "Distributed resource allocation through utility design - Part II: applications to submodular, supermodular and set covering problems". *ArXiv*, **2018**.