Covering Games:

Approximation through Non-Cooperation

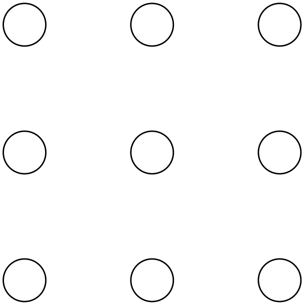
Martin Gairing

Warwick 2010

A general covering problem

Given

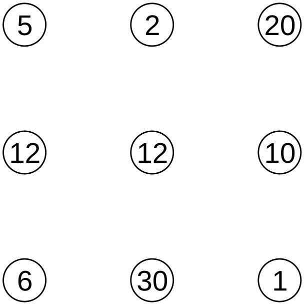
- a universe E of elements



A general covering problem

Given

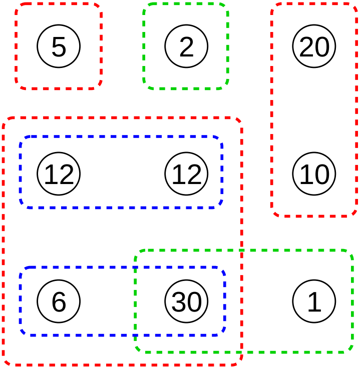
- a universe E of elements
- a weight function $w : E \mapsto \mathbb{N}$



A general covering problem

Given

- a universe E of elements
- a weight function $w : E \mapsto \mathbb{N}$
- n collections of subsets of E
  - $S_i \subset 2^E$ for each $i \in [n]$



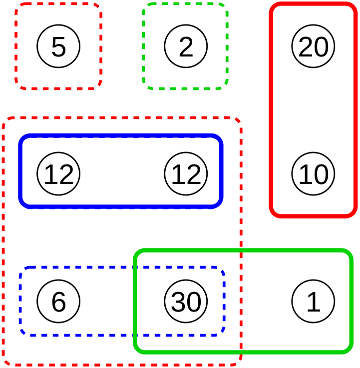
A general covering problem

Given

- a universe E of elements
- a weight function $w : E \mapsto \mathbb{N}$
- n collections of subsets of E
  - $S_i \subset 2^E$ for each $i \in [n]$

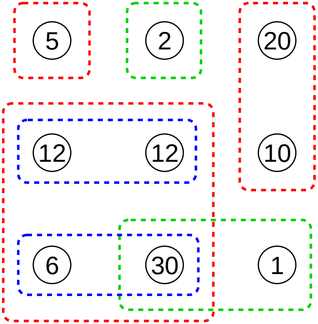
Task

- choose n subsets $(s_i)_{i \in [n]}$, s.t.
  - $s_i \in S_i$
  - $\bigcup_{i \in [n]} s_i$ has maximum total weight



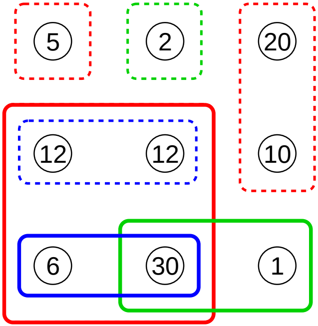
Covering Games

- *n* players



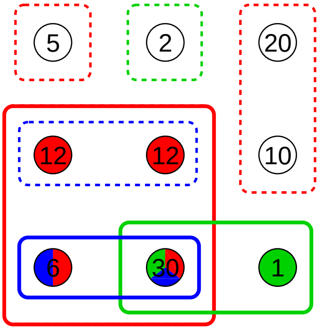
Covering Games

- n players
- player $i \in [n]$ chooses $s_i \in S_i$



Covering Games

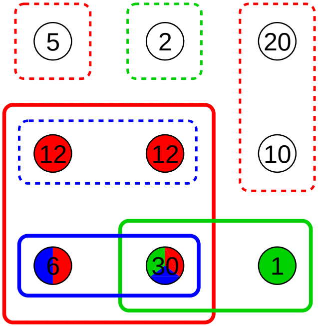
- n players
- player $i \in [n]$ chooses $s_i \in S_i$
- for covering an element, pay players according to utility sharing function
  - $f : [n] \mapsto [0, 1]$



$f(1) = 1 \quad f(2) = \frac{1}{2} \quad f(3) = \frac{1}{3}$

Covering Games

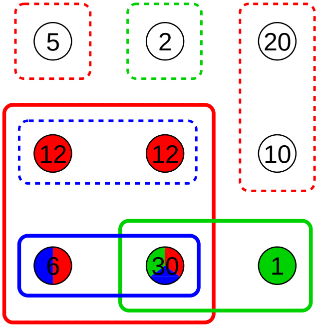
- n players
- player $i \in [n]$ chooses $s_i \in S_i$
- for covering an element, pay players according to utility sharing function
  - $f : [n] \mapsto [0, 1]$
- natural assumptions on f
  - non-increasing
  - no-overpay ($j \cdot f(j) \leq 1$)



$f(1) = 1 \qquad f(2) = \frac{1}{2} \qquad f(3) = \frac{1}{3}$

Covering Games

- n players
- player $i \in [n]$ chooses $s_i \in S_i$
- for covering an element, pay players according to utility sharing function
  - $f : [n] \mapsto [0, 1]$
- natural assumptions on f
  - non-increasing
  - no-overpay ($j \cdot f(j) \leq 1$)
- Load on $e \in E$:
  $$\delta_e(s) = |\{i \in [n] : e \in s_i\}|$$
- Utility of player $i \in [n]$:
  $$u_i(s) = \sum_{e \in s_i} f(\delta_e(s)) \cdot w_e$$



$f(1) = 1 \quad f(2) = \frac{1}{2} \quad f(3) = \frac{1}{3}$

- $u_1(s) = 37$
- $u_2(s) = 13$
- $u_3(s) = 11$

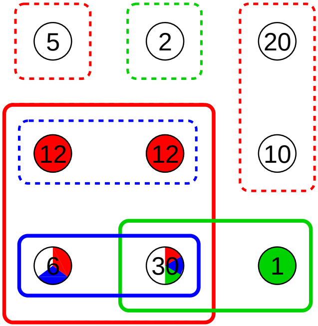
Covering Games

- n players
- player $i \in [n]$ chooses $s_i \in S_i$
- for covering an element, pay players according to utility sharing function
  - $f : [n] \mapsto [0, 1]$
- natural assumptions on f
  - non-increasing
  - no-overpay ($j \cdot f(j) \leq 1$)
- Load on $e \in E$:

$$\delta_e(\mathbf{s}) = |\{i \in [n] : e \in s_i\}|$$

- Utility of player $i \in [n]$:

$$u_i(\mathbf{s}) = \sum_{e \in s_i} f(\delta_e(\mathbf{s})) \cdot w_e$$



$f(1) = 1 \qquad f(2) = \frac{1}{3} \qquad f(3) = \frac{1}{6}$

- $u_1(\mathbf{s}) = 31$
- $u_2(\mathbf{s}) = 7$:-(
- $u_3(\mathbf{s}) = 6$

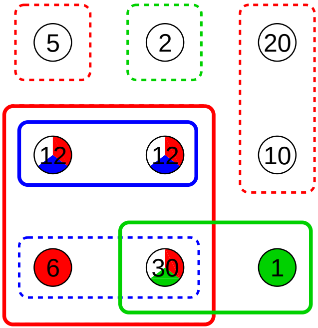
Covering Games

- n players
- player $i \in [n]$ chooses $s_i \in S_i$
- for covering an element, pay players according to utility sharing function
  - $f : [n] \mapsto [0,1]$
- natural assumptions on f
  - non-increasing
  - no-overpay ($j \cdot f(j) \leq 1$)
- Load on $e \in E$:

$$\delta_e(s) = |\{i \in [n] : e \in s_i\}|$$

- Utility of player $i \in [n]$:

$$u_i(s) = \sum_{e \in s_i} f(\delta_e(s)) \cdot w_e$$



$f(1) = 1 \qquad f(2) = \frac{1}{3} \qquad f(3) = \frac{1}{6}$

- $u_1(s) = 24$:-(
- $u_2(s) = 8$
- $u_3(s) = 11$

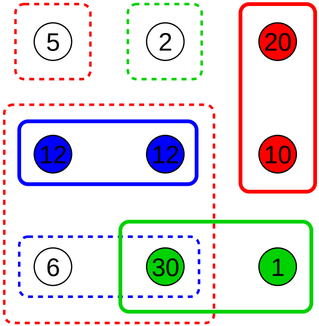
Covering Games

- n players
- player $i \in [n]$ chooses $s_i \in S_i$
- for covering an element, pay players according to utility sharing function
  - $f : [n] \mapsto [0, 1]$
- natural assumptions on f
  - non-increasing
  - no-overpay ($j \cdot f(j) \leq 1$)
- Load on $e \in E$:

$$\delta_e(\mathrm{s}) = |\{i \in [n] : e \in s_i\}|$$

- Utility of player $i \in [n]$:

$$u_i(\mathrm{s}) = \sum_{e \in s_i} f(\delta_e(\mathrm{s})) \cdot w_e$$



$f(1) = 1 \qquad f(2) = \frac{1}{3} \qquad f(3) = \frac{1}{6}$

- $u_1(\mathrm{s}) = 30$
- $u_2(\mathrm{s}) = 24$
- $u_3(\mathrm{s}) = 31$

Special Cases

MAX-k-Cover

$S_i = S_j$ for all players $i, j \in [n]$

[NEMHAUSER, WOLSEY, FISHER, '78]
Greedy $\Rightarrow (1 - \frac{1}{e}) - \mathit{approx}$.

[FEIGE, '98]
better $\Rightarrow NP \subseteq TIME(n^{O(\log \log n)})$

Special Cases

MAX-k-Cover

$S_i = S_j$ for all players $i, j \in [n]$

[Nemhauser, Wolsey, Fisher, '78]

Greedy $\Rightarrow (1 - \frac{1}{e}) - approx.$

[Feige, '98]

better $\Rightarrow NP \subseteq TIME(n^{O(\log \log n)})$

SAT-Games

$|S_i| \leq 2$ for each player $i \in [n]$

[Giannakos et al., '07]

Special Cases

MAX-k-Cover

$S_i = S_j$ for all players $i, j \in [n]$

[NEMHAUSER, WOLSEY, FISHER, '78]

Greedy $\Rightarrow (1 - \frac{1}{e}) - approx.$

[FEIGE, '98]

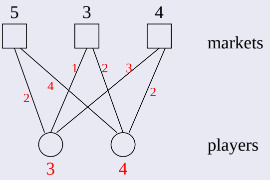
better $\Rightarrow NP \subseteq TIME(n^{O(\log \log n)})$

SAT-Games

$|S_i| \leq 2$ for each player $i \in [n]$

[GIANNAKOS ET AL., '07]

Market-Sharing Games



[GOEMANS, MIRROKNI, THOTTAN, '04]

Nash Equilibrium

Nash Equilibrium

The (pure) strategy profile s is a pure Nash equilibrium if and only if all players $i \in [n]$ are satisfied, that is,

$$u_i(\mathrm{s}) \geq u_i(\mathrm{s}_{-i}, s_i') \qquad \text{for all } i \in [n] \text{ and } s_i' \in S_i.$$

Nash Equilibrium

Nash Equilibrium

The (pure) strategy profile s is a pure Nash equilibrium if and only if all players $i \in [n]$ are satisfied, that is,

$$u_i(\mathrm{s}) \geq u_i(\mathrm{s}_{-i}, s_i') \qquad \text{for all } i \in [n] \text{ and } s_i' \in S_i.$$

Proposition [Rosenthal, 1973]

Every covering game admits a pure Nash equilibrium.

Rosenthals potential function:

$$\Phi(\mathrm{s}) = \sum_{e \in E} \sum_{i=1}^{\delta_e(\mathrm{s})} f_e(i)$$

If a single player increases her payoff by Δ then also the potential increases by Δ.

Price of Anarchy

- $W(s)$ . . . total weight of elements covered in s
- f . . . utility sharing function.

Price of Anarchy

$$\mathrm{PoA}_f = \inf_{\substack{\Gamma \in \mathcal{G}, \\ s \text{ is NE in } \Gamma}} \frac{W(s)}{\mathrm{OPT}}$$

Price of Anarchy

- $W(s)$ ... total weight of elements covered in s
- f ... utility sharing function.

Price of Anarchy

$$\mathrm{PoA}_f = \inf_{\substack{\Gamma \in \mathcal{G}, \\ s \text{ is NE in } \Gamma}} \frac{W(s)}{\mathrm{OPT}}$$

Main task

Construct utility sharing function that maximizes PoA_f.

Price of Anarchy

- $W(s)$ ... total weight of elements covered in s
- f ... utility sharing function.

Price of Anarchy

$$\text{PoA}_f = \inf_{\substack{\Gamma \in \mathcal{G}, \\ s \text{ is NE in } \Gamma}} \frac{W(s)}{\text{OPT}}$$

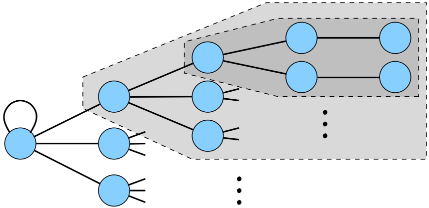
Main task

Construct utility sharing function that maximizes PoA_f.

- Coordination Mechanism [Christodoulou, Koutsoupias, Nanavati, '04]

What to hope for?

Example: k=4

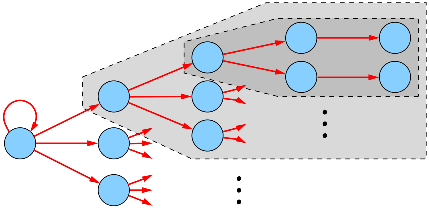


- node $\leftrightarrow$ element ($w_e = 1$)
- edge $\leftrightarrow$ player
  - $|S_1| = 1$
  - $|S_i| = 2$ for $i \geq 2$
- $k + 1$ levels
  - root: $k - 1$ children
  - level j node: $k - j$ children

$f : [n] \mapsto \mathbb{R}$ depends only on the number of players choosing an element.

What to hope for?

Example: k=4



- node $\leftrightarrow$ element ($w_e = 1$)
- edge $\leftrightarrow$ player
  - $|S_1| = 1$
  - $|S_i| = 2$ for $i \geq 2$
- $k + 1$ levels
  - root: $k - 1$ children
  - level j node: $k - j$ children

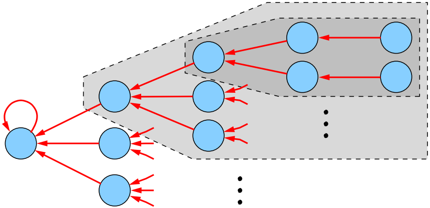
$f : [n] \mapsto \mathbb{R}$ depends only on the number of players choosing an element.

Optimum s*

$$W(s^*) = 1 + \sum_{j=1}^{k} (k-1) \cdot \frac{(k-1)!}{(k-j)!}$$

What to hope for?

Example: k=4



- node $\leftrightarrow$ element ($w_e = 1$)
- edge $\leftrightarrow$ player
  - $|S_1| = 1$
  - $|S_i| = 2$ for $i \geq 2$
- $k + 1$ levels
  - root: $k - 1$ children
  - level j node: $k - j$ children

$f : [n] \mapsto \mathbb{R}$ depends only on the number of players choosing an element.

Nash Equilibrium s

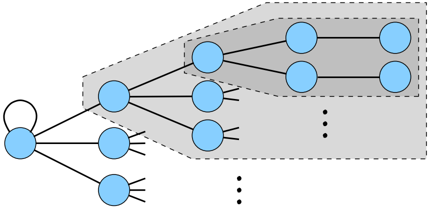
$W(s) = 1 + \sum_{j=1}^{k-1} (k-1) \cdot \frac{(k-1)!}{(k-j)!}$

Optimum s*

$W(s^*) = 1 + \sum_{j=1}^{k} (k-1) \cdot \frac{(k-1)!}{(k-j)!}$

What to hope for?

Example: k=4



- node $\leftrightarrow$ element ($w_e = 1$)
- edge $\leftrightarrow$ player
  - $|S_1| = 1$
  - $|S_i| = 2$ for $i \geq 2$
- $k + 1$ levels
  - root: $k - 1$ children
  - level j node: $k - j$ children

$f : [n] \mapsto \mathbb{R}$ depends only on the number of players choosing an element.

Nash Equilibrium s

$$W(s) = 1 + \sum_{j=1}^{k-1} (k-1) \cdot \frac{(k-1)!}{(k-j)!}$$

Optimum s*

$$W(s^*) = 1 + \sum_{j=1}^{k} (k-1) \cdot \frac{(k-1)!}{(k-j)!}$$

Theorem

$$\text{PoA}_f(k) \leq 1 - \frac{1}{\frac{1}{(k-1)(k-1)!} + \sum_{j=0}^{k-1} \frac{1}{j!}}.$$

What is known?

A simple example shows:

- If f is defined by $f(j) = \frac{1}{j}$ for all $j \in \mathbb{N}$

  $\Rightarrow \mathrm{PoA}_f \leq \frac{1}{2}$

What is known?

A simple example shows:

- If f is defined by $f(j) = \frac{1}{j}$ for all $j \in \mathbb{N}$
  $\Rightarrow \text{PoA}_f \leq \frac{1}{2}$

Consider utility sharing function which is

- non-increasing,
- $j \cdot f(j) \leq 1$ (no-overpay), and
- $f(1) = 1$

Then the covering game is also a valid utility game.

What is known?

A simple example shows:

- If f is defined by $f(j) = \frac{1}{j}$ for all $j \in \mathbb{N}$
  $\Rightarrow \mathrm{PoA}_f \leq \frac{1}{2}$

Consider utility sharing function which is

- non-increasing,
- $j \cdot f(j) \leq 1$ (no-overpay), and
- $f(1) = 1$

Then the covering game is also a valid utility game.

Theorem [VETTA, '02]

$$\mathrm{PoA}_f \geq \frac{1}{2}$$

General Lower Bound on PoA:

- Given utility sharing function f
- Define $\chi = \chi(f)$ as the smallest number, such that $\forall j \in \mathbb{N}$:

$$j \cdot f(j) - f(j+1) \leq \chi \cdot f(1)$$

General Lower Bound on PoA:

- Given utility sharing function f
- Define $\chi = \chi(f)$ as the smallest number, such that $\forall j \in \mathbb{N}$:

$$j \cdot f(j) - f(j+1) \leq \chi \cdot f(1)$$

Theorem

$$\text{PoA}_f \geq \frac{1}{\chi + 1}$$

General Lower Bound on PoA:

- Given utility sharing function f
- Define $\chi = \chi(f)$ as the smallest number, such that $\forall j \in \mathbb{N}$:

$$j \cdot f(j) - f(j+1) \leq \chi \cdot f(1)$$

Theorem

$$\text{PoA}_f \geq \frac{1}{\chi + 1}$$

Remarks

- Construct f such that χ is minimized.

Construct f that minimizes χ

Task

min χ s.t.

- $i \cdot f(i) - f(i+1) \leq \chi \cdot f(1) \quad$ for all $i \in [k-1]$
- $(k-1) \cdot f(k) \leq \chi \cdot f(1)$

Construct f that minimizes χ

Task

min χ s.t.

- $i \cdot f(i) - f(i+1) \leq \chi \cdot f(1)$ for all $i \in [k-1]$
- $(k-1) \cdot f(k) \leq \chi \cdot f(1)$

$$\begin{pmatrix} 1-\chi & -1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ -\chi & 2 & -1 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & & & & \vdots \\ -\chi & 0 & \cdots & i & -1 & 0 & \cdots & 0 \\ \vdots & & & & \ddots & \ddots & & \vdots \\ -\chi & 0 & \cdots & 0 & 0 & k-2 & -1 & 0 \\ -\chi & 0 & \cdots & 0 & 0 & 0 & k-1 & -1 \\ -\chi & 0 & \cdots & 0 & 0 & 0 & 0 & k-1 \end{pmatrix}$$

Construct f that minimizes χ

Task

min χ s.t.

- $i \cdot f(i) - f(i+1) \leq \chi \cdot f(1)$ for all $i \in [k-1]$
- $(k-1) \cdot f(k) \leq \chi \cdot f(1)$

$$
\begin{pmatrix}
1-\chi & -1 & 0 & 0 & 0 & 0 & \cdots & 0 \\
-\chi & 2 & -1 & 0 & 0 & 0 & \cdots & 0 \\
\vdots & & \ddots & \ddots & & & & \vdots \\
-\chi & 0 & \cdots & i & -1 & 0 & \cdots & 0 \\
\vdots & & & & \ddots & \ddots & & \vdots \\
-\chi & 0 & \cdots & 0 & 0 & k-2 & -1 & 0 \\
-\chi[1+(k-1)] & 0 & \cdots & 0 & 0 & 0 & (k-1)^2 & 0 \\
-\chi & 0 & \cdots & 0 & 0 & 0 & 0 & k-1
\end{pmatrix}
$$

Construct f that minimizes χ

Task

min χ s.t.

- $i \cdot f(i) - f(i+1) \leq \chi \cdot f(1)$ for all $i \in [k-1]$
- $(k-1) \cdot f(k) \leq \chi \cdot f(1)$

$$\begin{pmatrix}
1-\chi & -1 & 0 & 0 & 0 & 0 & \cdots & 0 \\
-\chi & 2 & -1 & 0 & 0 & 0 & \cdots & 0 \\
\vdots & & \ddots & \ddots & & & & \vdots \\
-\chi & 0 & \cdots & i & -1 & 0 & \cdots & 0 \\
\vdots & & & & \ddots & \ddots & & \vdots \\
-\chi\left[1+(k-1)\sum_{j=1}^{2}\frac{(k-1)!}{(k-j)!}\right] & 0 & \cdots & 0 & 0 & (k-1)^2(k-2) & 0 & 0 \\
-\chi[1+(k-1)] & 0 & \cdots & 0 & 0 & 0 & (k-1)^2 & 0 \\
-\chi & 0 & \cdots & 0 & 0 & 0 & 0 & k-1
\end{pmatrix}$$

Construct f that minimizes χ

Task

min χ s.t.

- $i \cdot f(i) - f(i+1) \leq \chi \cdot f(1)$ for all $i \in [k-1]$
- $(k-1) \cdot f(k) \leq \chi \cdot f(1)$

$$\begin{pmatrix}
(k-1)(k-1)! - \chi\left[1 + (k-1)\sum_{j=1}^{k-1} \frac{(k-1)!}{(k-j)!}\right] & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
\vdots & & \ddots & \ddots & & & & \vdots \\
-\chi\left[1 + (k-1)\sum_{j=1}^{k-i} \frac{(k-1)!}{(k-j)!}\right] & 0 & \cdots & (k-1)\frac{(k-1)!}{(i-1)!} & 0 & 0 & \cdots & 0 \\
\vdots & & & & \ddots & \ddots & & \vdots \\
-\chi[1 + (k-1)] & 0 & \cdots & 0 & 0 & 0 & (k-1)^2 & 0 \\
-\chi & 0 & \cdots & 0 & 0 & 0 & 0 & k-1
\end{pmatrix}$$

Construct f that minimizes χ

$$\begin{pmatrix} (k-1)(k-1)! - \chi\left[1+(k-1)\sum_{j=1}^{k-1}\frac{(k-1)!}{(k-j)!}\right] & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & & & & \vdots \\ -\chi\left[1+(k-1)\sum_{j=1}^{k-i}\frac{(k-1)!}{(k-j)!}\right] & 0 & \cdots & (k-1)\frac{(k-1)!}{(i-1)!} & 0 & 0 & \cdots & 0 \\ \vdots & & & & \ddots & \ddots & & \vdots \\ -\chi[1+(k-1)] & 0 & \cdots & 0 & 0 & 0 & (k-1)^2 & 0 \\ -\chi & 0 & \cdots & 0 & 0 & 0 & 0 & k-1 \end{pmatrix}$$

k is known

- $\chi = \dfrac{1}{\frac{1}{(k-1)(k-1)!} + \sum_{j=1}^{k-1}\frac{1}{j!}}$

Construct f that minimizes χ

$$\begin{pmatrix}
(k-1)(k-1)! - \chi\left[1+(k-1)\sum_{j=1}^{k-1}\frac{(k-1)!}{(k-j)!}\right] & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
\vdots & & \ddots & \ddots & & & & \vdots \\
-\chi\left[1+(k-1)\sum_{j=1}^{k-i}\frac{(k-1)!}{(k-j)!}\right] & 0 & \cdots & (k-1)\frac{(k-1)!}{(i-1)!} & 0 & 0 & \cdots & 0 \\
\vdots & & & & \ddots & \ddots & & \vdots \\
-\chi[1+(k-1)] & 0 & \cdots & 0 & 0 & 0 & (k-1)^2 & 0 \\
-\chi & 0 & \cdots & 0 & 0 & 0 & 0 & k-1
\end{pmatrix}$$

k is known

- $\chi = \dfrac{1}{\frac{1}{(k-1)(k-1)!} + \sum_{j=1}^{k-1}\frac{1}{j!}}$
- Utility sharing function:

$$f(i) = (i-1)!\,\frac{\frac{1}{(k-1)(k-1)!} + \sum_{j=i}^{k-1}\frac{1}{j!}}{\frac{1}{(k-1)(k-1)!} + \sum_{j=1}^{k-1}\frac{1}{j!}}$$

Construct f that minimizes χ

$$\begin{pmatrix}
(k-1)(k-1)! - \chi\left[1+(k-1)\sum_{j=1}^{k-1}\frac{(k-1)!}{(k-j)!}\right] & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
\vdots & & \ddots & \ddots & & & & \vdots \\
-\chi\left[1+(k-1)\sum_{j=1}^{k-i}\frac{(k-1)!}{(k-j)!}\right] & 0 & \cdots & (k-1)\frac{(k-1)!}{(i-1)!} & 0 & 0 & \cdots & 0 \\
\vdots & & & & \ddots & \ddots & & \vdots \\
-\chi[1+(k-1)] & 0 & \cdots & 0 & 0 & 0 & (k-1)^2 & 0 \\
-\chi & 0 & \cdots & 0 & 0 & 0 & 0 & k-1
\end{pmatrix}$$

k is known

- $\chi = \frac{1}{\frac{1}{(k-1)(k-1)!}+\sum_{j=1}^{k-1}\frac{1}{j!}}$
- Utility sharing function:

  $$f(i) = (i-1)!\,\frac{\frac{1}{(k-1)(k-1)!}+\sum_{j=i}^{k-1}\frac{1}{j!}}{\frac{1}{(k-1)(k-1)!}+\sum_{j=1}^{k-1}\frac{1}{j!}}$$
- $\mathrm{PoA}_f \geq 1 - \frac{1}{\frac{1}{(k-1)(k-1)!}+\sum_{j=0}^{k-1}\frac{1}{j!}}$

Construct f that minimizes χ

$$\begin{pmatrix} (k-1)(k-1)! - \chi\left[1 + (k-1)\sum_{j=1}^{k-1} \frac{(k-1)!}{(k-j)!}\right] & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & & & & \vdots \\ -\chi\left[1 + (k-1)\sum_{j=1}^{k-i} \frac{(k-1)!}{(k-j)!}\right] & 0 & \cdots & (k-1)\frac{(k-1)!}{(i-1)!} & 0 & 0 & \cdots & 0 \\ \vdots & & & & \ddots & \ddots & & \vdots \\ -\chi[1 + (k-1)] & 0 & \cdots & 0 & 0 & 0 & (k-1)^2 & 0 \\ -\chi & 0 & \cdots & 0 & 0 & 0 & 0 & k-1 \end{pmatrix}$$

k is known

- $\chi = \frac{1}{\frac{1}{(k-1)(k-1)!} + \sum_{j=1}^{k-1} \frac{1}{j!}}$
- Utility sharing function:

  $f(i) = (i-1)! \frac{\frac{1}{(k-1)(k-1)!} + \sum_{j=i}^{k-1} \frac{1}{j!}}{\frac{1}{(k-1)(k-1)!} + \sum_{j=1}^{k-1} \frac{1}{j!}}$
- $\mathrm{PoA}_f \geq 1 - \frac{1}{\frac{1}{(k-1)(k-1)!} + \sum_{j=0}^{k-1} \frac{1}{j!}}$

k is unknown ($k \to \infty$)

- $\chi = \frac{1}{e-1}$
- Utility sharing function:

  $f(i) = (i-1)! \frac{e - \sum_{j=0}^{i-1} \frac{1}{j!}}{e-1}$
- $\mathrm{PoA}_f \geq 1 - \frac{1}{e}$

Distributed Approximation Algorithm

Task

- Turn this into $(1 - \frac{1}{e})$-approximation algorithm.
  - Start with arbitrary strategy profile.
  - Let players unilaterally improve. (selfish steps)
- Use Rosenthals potential function to bound running time.

Distributed Approximation Algorithm

Task

- Turn this into $(1 - \frac{1}{e})$-approximation algorithm.
  - Start with arbitrary strategy profile.
  - Let players unilaterally improve. (selfish steps)
- Use Rosenthals potential function to bound running time.

Problem

- Increase in potential function can be arbitrary small.

Distributed Approximation Algorithm

Task

- Turn this into $(1 - \frac{1}{e})$-approximation algorithm.
  - Start with arbitrary strategy profile.
  - Let players unilaterally improve. (selfish steps)
- Use Rosenthals potential function to bound running time.

Problem

- Increase in potential function can be arbitrary small.

Solution

- choose constant $k' \in \mathbb{N}$
- $f(i) = 0$ for $i > k'$
- This yields $(1 - \frac{1}{e} - \varepsilon)$-approximation algorithm $(\varepsilon = \varepsilon(k') = o(1))$

A local search approximation algorithm

Theorem

For every $\varepsilon > 0$, there exists a (local-search) approximation algorithm

- with approximation ratio $1 - \frac{1}{e} - \varepsilon$,
- that uses at most $\mathcal{O}(\frac{1}{\varepsilon} \cdot \log\log(\frac{1}{\varepsilon})) \cdot W$ selfish steps.

Best Possible [Feige, JouACM'98]

A local search approximation algorithm

Theorem

For every $\varepsilon > 0$, there exists a (local-search) approximation algorithm

- with approximation ratio $1 - \frac{1}{e} - \varepsilon$,
- that uses at most $\mathcal{O}(\frac{1}{\varepsilon} \cdot \log\log(\frac{1}{\varepsilon})) \cdot W$ selfish steps.

- What happens if W is arbitrary?

Theorem

Then, for every (non-constant) utility sharing function, computing a pure Nash equilibrium is PLS-complete.

A local search approximation algorithm

Theorem

For every $\varepsilon > 0$, there exists a (local-search) approximation algorithm

- with approximation ratio $1 - \frac{1}{e} - \varepsilon$,
- that uses at most $\mathcal{O}(\frac{1}{\varepsilon} \cdot \log\log(\frac{1}{\varepsilon})) \cdot W$ selfish steps.

- What happens if W is arbitrary?

Theorem

Then, for every (non-constant) utility sharing function, computing a pure Nash equilibrium is PLS-complete.

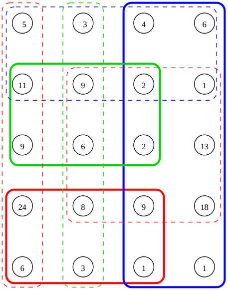
Theorem

There exists a (centralized) polynomial-time $(1 - \frac{1}{e})$-approximation algorithm for the general covering problem.

Covering Games

We showed:

- For every utility sharing function f, $\text{PoA}_f \leq 1 - \frac{1}{e}$.
- There exists f with $\text{PoA}_f \geq 1 - \frac{1}{e}$.
- Local search approximation algorithm if W is bounded by polynom in $n, |E|$.
- Limits of our approach
- Centralized Approximation Algorithm



Covering Games

We showed:

- For every utility sharing function f, $\mathrm{PoA}_f \leq 1 - \frac{1}{e}$.
- There exists f with $\mathrm{PoA}_f \geq 1 - \frac{1}{e}$.
- Local search approximation algorithm if W is bounded by polynom in $n, |E|$.
- Limits of our approach
- Centralized Approximation Algorithm

Open Problems

- weighted case: restrict to ε-NE
- More general models
  - w_e is not constant
  - element must be covered multiple times
  - ...