

Sustainability in Multi-Agent Systems

Filippo Bistaffa (IIIA-CSIC)



July 6-7, 2022

Computational Sustainability via Cooperation

Team Formation



Collective Energy Purchasing



Shared Mobility



Formation of Optimal Collectives

Computational Sustainability via Cooperation

Team Formation



Collective Energy Purchasing



Shared Mobility



Formation of Optimal Collectives

Constrained Optimisation for Sustainability


Constrained Optimisation as Theoretical Framework

Constrained optimisation (**fundamental area** of AI) used as technique to achieve **computational sustainability** via **optimal collective formation**

Challenge in Real-World Scenarios


The number of possible collectives is *exponential* (“curse of dimensionality”), so large-scale optimisation problems are *computationally very hard* to solve

Agenda

 July 6, 14:30 – 15:30:

Theoretical Foundations of Constrained Optimisation in MAS

Practical Applications of Constrained Optimisation in MAS

 July 7, 9:30 – 10:30:

Google Colab Hands-On Session

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

- Combinatorial Auctions

- Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

- Ridesharing (Computational Challenge)

- Team Formation (Modelling Challenge)

Google Colab Hands-On Session

- Induced Subgraph Games

- Approximately Equivalent ISGs

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

- Combinatorial Auctions

- Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

- Ridesharing (Computational Challenge)

- Team Formation (Modelling Challenge)

Google Colab Hands-On Session

- Induced Subgraph Games

- Approximately Equivalent ISGs

Single-Item Auctions



Winner Determination Problem (WDP)

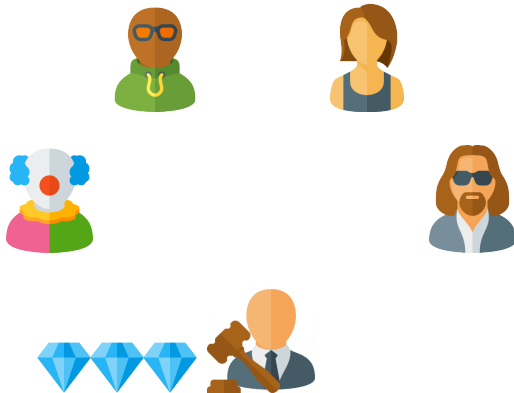
Objective

Given a set of bids, allocate the good to the bidder whose bid *maximises* the auctioneer's revenue

WDPs for Single-Item Auctions are Easy

- English: last bid wins
- Japanese: last remaining bidder wins
- Dutch: first bid wins

Multi-Unit Auctions



WDP for Multi-Unit Auctions

Example of a Multi-Unit Auction

We want to sell 15 apples maximising the revenue

What is the Optimal Allocation with these Bids?

- A: buy 12 apples for 4€ $[V_A(\{\text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}\}) = 4\text{€}]$
- B: buy 2 apples for 2€ $[V_B(\{\text{🍏}, \text{🍏}\}) = 2\text{€}]$
- C: buy 1 apple for 2€ $[V_C(\{\text{🍏}\}) = 2\text{€}]$
- D: buy 1 apple for 1€ $[V_D(\{\text{🍏}\}) = 1\text{€}]$
- E: buy 4 apples for 10€ $[V_E(\{\text{🍏}, \text{🍏}, \text{🍏}, \text{🍏}\}) = 10\text{€}]$

WDP as a Weighted Knapsack Problem

- Let x_A, x_B, x_C, x_D, x_E be decision variables [One binary variable for each bid]
- Maximise the revenue obtained by filling the backpack

Integer *Linear* Programming (ILP) Formulation

$$\begin{array}{ll}\text{maximise} & 4 \cdot x_A + 2 \cdot x_B + 2 \cdot x_C + x_D + 10 \cdot x_E \quad [\text{Values of accepted bids}] \\ \text{subject to} & 12 \cdot x_A + 2 \cdot x_B + x_C + x_D + 4 \cdot x_E \leq 15 \quad [\text{"Capacity" constraint}] \\ & x_A, x_B, x_C, x_D, x_E \in \{0, 1\} \quad [\text{Binary decision variables}]\end{array}$$

WDP as a Weighted Knapsack Problem

- Let x_A, x_B, x_C, x_D, x_E be decision variables [One binary variable for each bid]
- Maximise the revenue obtained by filling the backpack

Integer *Linear* Programming (ILP) Formulation

$$\begin{array}{ll}\text{maximise} & 4 \cdot x_A + 2 \cdot x_B + 2 \cdot x_C + x_D + 10 \cdot x_E \quad [\text{Values of accepted bids}] \\ \text{subject to} & 12 \cdot x_A + 2 \cdot x_B + x_C + x_D + 4 \cdot x_E \leq 15 \quad [\text{"Capacity" constraint}] \\ & x_A, x_B, x_C, x_D, x_E \in \{0, 1\} \quad [\text{Binary decision variables}]\end{array}$$

WDP as a Weighted Knapsack Problem

- Let x_A, x_B, x_C, x_D, x_E be decision variables [One binary variable for each bid]
- Maximise the revenue obtained by filling the backpack

Integer *Linear* Programming (ILP) Formulation

$$\begin{array}{ll}\text{maximise} & 4 \cdot x_A + 2 \cdot x_B + 2 \cdot x_C + x_D + 10 \cdot x_E \quad [\text{Values of accepted bids}] \\ \text{subject to} & 12 \cdot x_A + 2 \cdot x_B + x_C + x_D + 4 \cdot x_E \leq 15 \quad [\text{"Capacity" constraint}] \\ & x_A, x_B, x_C, x_D, x_E \in \{0, 1\} \quad [\text{Binary decision variables}]\end{array}$$

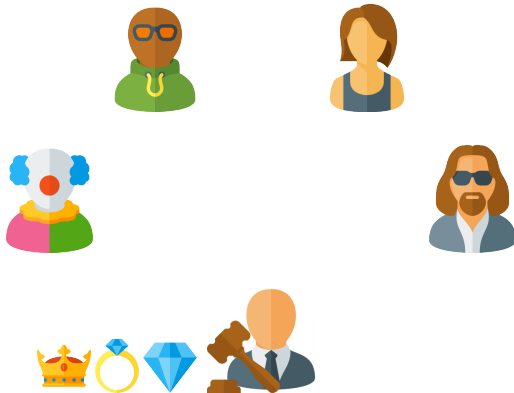
WDP as a Weighted Knapsack Problem

- Let x_A, x_B, x_C, x_D, x_E be decision variables [One binary variable for each bid]
- Maximise the revenue obtained by filling the backpack

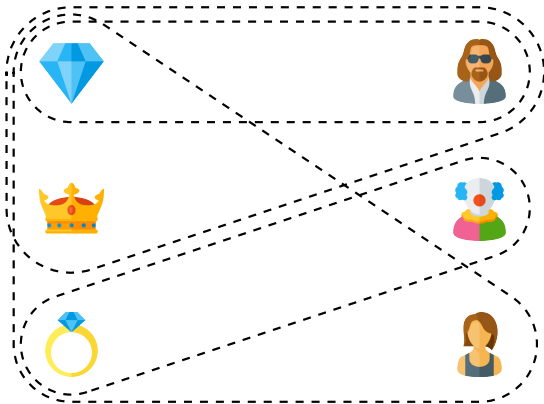
Integer *Linear* Programming (ILP) Formulation

$$\begin{array}{ll}\text{maximise} & 4 \cdot x_A + 2 \cdot x_B + 2 \cdot x_C + x_D + 10 \cdot x_E \quad [\text{Values of accepted bids}] \\ \text{subject to} & 12 \cdot x_A + 2 \cdot x_B + x_C + x_D + 4 \cdot x_E \leq 15 \quad [\text{"Capacity" constraint}] \\ & x_A, x_B, x_C, x_D, x_E \in \{0, 1\} \quad [\text{Binary decision variables}]\end{array}$$

Multi-Item (Combinatorial) Auctions



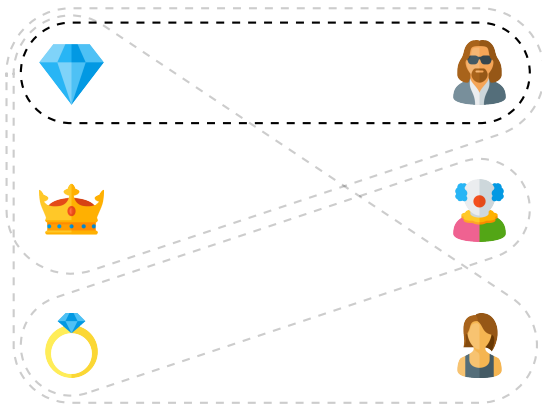
Multi-Item (Combinatorial) Auctions



Multi-item bids

- $V_{\text{Man}}(\{\text{Diamond}\}) = 0\text{€}$
- $V_{\text{Man}}(\{\text{Diamond}, \text{Crown}\}) = 400\text{€}$
- $V_{\text{Woman 2}}(\{\text{Ring}\}) = 100\text{€}$
- $V_{\text{Woman 1}}(\{\text{Diamond}, \text{Ring}, \text{Crown}\}) = 450\text{€}$

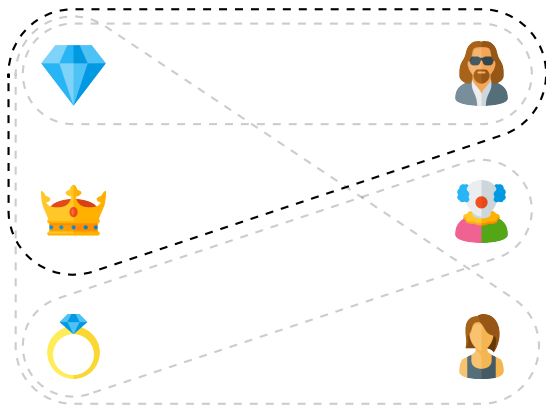
Multi-Item (Combinatorial) Auctions



Multi-item bids

- $V_{\text{man}}(\{\text{diamond}\}) = 0\text{€}$
- $V_{\text{man}}(\{\text{diamond}, \text{crown}\}) = 400\text{€}$
- $V_{\text{woman1}}(\{\text{ring}\}) = 100\text{€}$
- $V_{\text{woman2}}(\{\text{diamond}, \text{ring}, \text{crown}\}) = 450\text{€}$

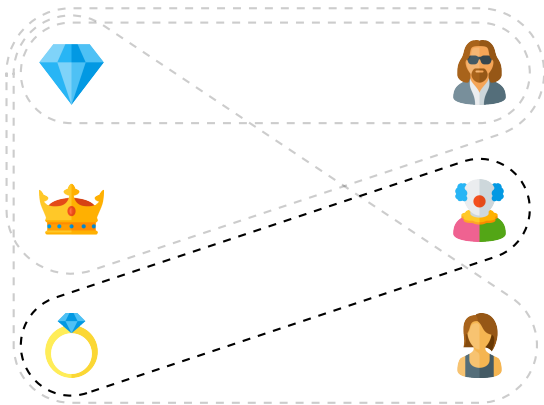
Multi-Item (Combinatorial) Auctions



Multi-item bids

- $V_{\text{Man}}(\{\text{Diamond}\}) = 0\text{€}$
- $V_{\text{Man}}(\{\text{Diamond}, \text{Crown}\}) = 400\text{€}$
- $V_{\text{Woman 1}}(\{\text{Ring}\}) = 100\text{€}$
- $V_{\text{Woman 2}}(\{\text{Diamond}, \text{Ring}, \text{Crown}\}) = 450\text{€}$

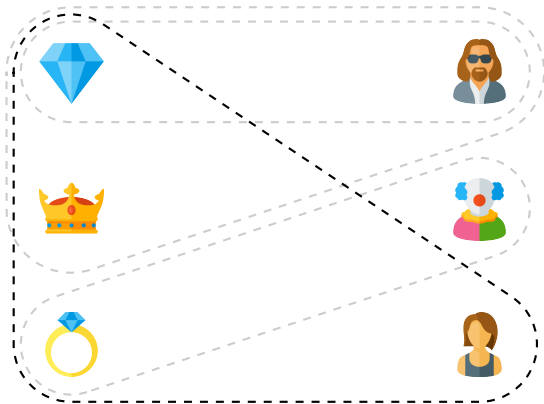
Multi-Item (Combinatorial) Auctions



Multi-item bids

- $V_{\text{man}}(\{\text{diamond}\}) = 0\text{€}$
- $V_{\text{man}}(\{\text{diamond}, \text{crown}\}) = 400\text{€}$
- $V_{\text{woman1}}(\{\text{diamond ring}\}) = 100\text{€}$
- $V_{\text{woman2}}(\{\text{diamond}, \text{diamond ring}, \text{crown}\}) = 450\text{€}$

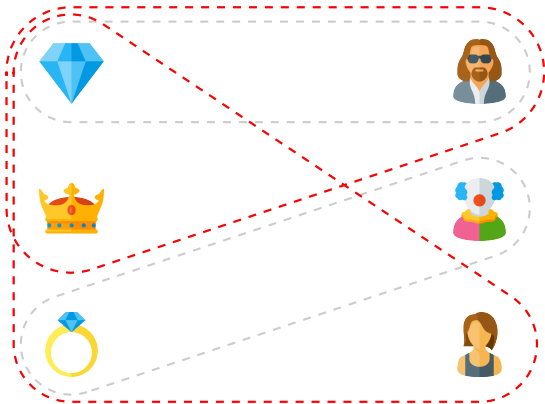
Multi-Item (Combinatorial) Auctions



Multi-item bids

- $V_{\text{man}}(\{\text{diamond}\}) = 0\text{€}$
- $V_{\text{man}}(\{\text{diamond}, \text{crown}\}) = 400\text{€}$
- $V_{\text{clown}}(\{\text{ring}\}) = 100\text{€}$
- $V_{\text{woman}}(\{\text{diamond}, \text{ring}, \text{crown}\}) = 450\text{€}$

Multi-Item (Combinatorial) Auctions



Multi-item bids

- $V_{\text{Man}}(\{\text{Diamond}\}) = 0\text{€}$
- $V_{\text{Man}}(\{\text{Diamond}, \text{Crown}\}) = 400\text{€}$
- $V_{\text{Clown}}(\{\text{Ring}\}) = 100\text{€}$
- $V_{\text{Woman}}(\{\text{Diamond}, \text{Ring}, \text{Crown}\}) = 450\text{€}$

WDP as Weighted Set Packing (WSP) Problem

- Given a set N of items and a set \mathcal{S} of bids, let M be a $|N| \times |\mathcal{S}|$ matrix
- $M_{iS} = 1$ if and only if item $i \in N$ is part of bid $S \in \mathcal{S}$, $M_{iS} = 0$ otherwise

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Diagram illustrating the matrix M and the items/bids it represents:

- Items (rows):
 - Item 1: Blue diamond
 - Item 2: Yellow crown
 - Item 3: Yellow ring
- Bids (columns):
 - Column 1: {Blue diamond}
 - Column 2: {Blue diamond, Yellow crown}
 - Column 3: {Blue diamond, Yellow ring}
 - Column 4: {Blue diamond, Yellow crown, Yellow ring}

Weighted Set Packing (WSP) Problem

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

ILP Formulation for WSP

$$\begin{array}{ll} \text{maximise} & \sum_{S \in \mathcal{S}} x_S \cdot V(S) \quad [\text{Value of each active bid}] \\ \text{subject to} & \sum_{S \in \mathcal{S}} M_{iS} \cdot x_S = 1 \quad \forall i \in N \quad [\text{All items must be sold}] \\ & \sum_{S \in \mathcal{S}} M_{iS} \cdot x_S \leq 1 \quad \forall i \in N \quad [\text{Items can remain unsold}] \end{array}$$

Weighted Set Packing (WSP) Problem

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

ILP Formulation for WSP

$$\begin{array}{ll} \text{maximise} & \sum_{S \in \mathcal{S}} x_S \cdot V(S) \quad [\text{Value of each active bid}] \\ \text{subject to} & \sum_{S \in \mathcal{S}} M_{iS} \cdot x_S = 1 \quad \forall i \in N \quad [\text{All items must be sold}] \\ & \sum_{S \in \mathcal{S}} M_{iS} \cdot x_S \leq 1 \quad \forall i \in N \quad [\text{Items can remain unsold}] \end{array}$$

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

Combinatorial Auctions

Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

Ridesharing (Computational Challenge)

Team Formation (Modelling Challenge)

Google Colab Hands-On Session

Induced Subgraph Games

Approximately Equivalent ISGs

Characteristic Function Games (CFGs)



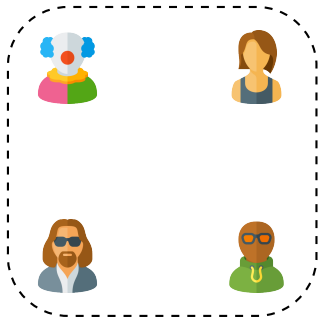
Set of Agents A

$$A = \{\text{robot}, \text{woman}, \text{man1}, \text{man2}\}$$

Characteristic Function $v(\cdot)$

- $v(\{\text{man1}, \text{woman}\}) = 0$
- $v(\{\text{man1}, \text{robot}, \text{man2}\}) = -7$
- $v(\{\text{man1}, \text{robot}\}) = 3$
- ...

Characteristic Function Games (CFGs)



Set of Agents A

$$A = \{\text{agent 1}, \text{agent 2}, \text{agent 3}, \text{agent 4}\}$$

Characteristic Function $v(\cdot)$

- $v(\{\text{agent 1}, \text{agent 3}\}) = 0$
- $v(\{\text{agent 2}, \text{agent 3}, \text{agent 4}\}) = -7$
- $v(\{\text{agent 1}, \text{agent 2}\}) = 3$
- ...

Characteristic Function Games (CFGs)



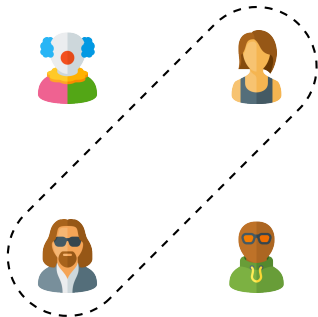
Set of Agents A

$$A = \{\text{robot}, \text{female}, \text{male1}, \text{male2}\}$$

Characteristic Function $v(\cdot)$

- $v(\{\text{male1}, \text{female}\}) = 0$
- $v(\{\text{male1}, \text{robot}, \text{male2}\}) = -7$
- $v(\{\text{male1}, \text{robot}\}) = 3$
- ...

Characteristic Function Games (CFGs)



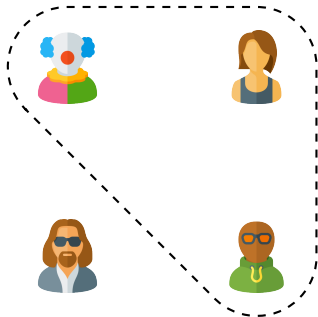
Set of Agents A

$$A = \{\text{man with sunglasses}, \text{clown}, \text{woman}, \text{man with glasses}\}$$

Characteristic Function $v(\cdot)$

- $v(\{\text{man with sunglasses}, \text{woman}\}) = 0$
- $v(\{\text{clown}, \text{woman}, \text{man with glasses}\}) = -7$
- $v(\{\text{man with sunglasses}, \text{clown}\}) = 3$
- ...

Characteristic Function Games (CFGs)



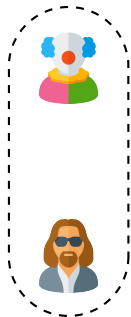
Set of Agents A

$$A = \{\text{👤}, \text{👤}, \text{👤}, \text{👤}\}$$

Characteristic Function $v(\cdot)$

- $v(\{\text{👤}, \text{👤}\}) = 0$
- $v(\{\text{👤}, \text{👤}, \text{👤}\}) = -7$
- $v(\{\text{👤}, \text{👤}\}) = 3$
- ...

Characteristic Function Games (CFGs)



Set of Agents A

$$A = \{\text{man with sunglasses}, \text{clown}, \text{woman}, \text{man with glasses}\}$$

Characteristic Function $v(\cdot)$

- $v(\{\text{man with sunglasses}, \text{woman}\}) = 0$
- $v(\{\text{woman}, \text{clown}, \text{man with glasses}\}) = -7$
- $v(\{\text{man with sunglasses}, \text{clown}\}) = 3$
- ...

Characteristic Function Games (CFGs)



Set of Agents A

$$A = \{\text{👨🏻, 👩🏻, 👨🏻, 👨🏻}\}$$

Characteristic Function $v(\cdot)$

- $v(\{\text{👨🏻, 👩🏻}\}) = 0$
- $v(\{\text{👨🏻, 👩🏻, 👨🏻}\}) = -7$
- $v(\{\text{👨🏻, 👩🏻}\}) = 3$
- ...

Coalition Structure Generation (CSG) \approx WDP for CFGs

Objective of Coalition Structure Generation

Compute the *partition* of A that *maximises* the sum of the corresponding values

ILP Formulation for Coalition Structure Generation

$$\begin{array}{ll} \text{maximise} & \sum_{S \in \mathcal{S}} x_S \cdot V(S) \quad [\text{Value of each selected coalition}] \\ \text{subject to} & \sum_{S \in \mathcal{S}} M_{iS} \cdot x_S = 1 \quad \forall i \in N \quad [\text{Each agent in } \textit{one} \text{ coalition}] \end{array}$$

Coalition Structure Generation (CSG) \approx WDP for CFGs

- Given A and a set \mathcal{S} of *coalitions* (i.e., subsets) of A , let M be a $|A| \times |\mathcal{S}|$ matrix
- $M_{iS} = 1$ if and only if agent $a \in A$ is part of coalition $S \in \mathcal{S}$, $M_{iS} = 0$ otherwise

$$M = \begin{matrix} & \begin{matrix} \{a,b\} & \{a,c\} & \{b,c\} & \{a,b,c\} & \{a,b\} & \{a,c\} & \{b,c\} & \{a,b,c\} \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} & \begin{matrix} \text{Agent } a \\ \text{Agent } b \\ \text{Agent } c \end{matrix} \end{matrix}$$

Characteristic Function

Characteristic Function

The function $v : \mathcal{P}(A) \rightarrow \mathbb{R}$ associates a value to *every coalition* (i.e., subset) of A

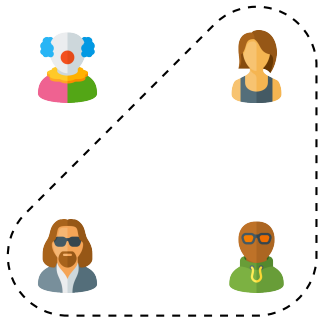
Exponential Complexity

Representing $v(\cdot)$ as a *table* requires an *exponential* number of steps (i.e., $2^{|A|}$)

Mitigate this Complexity

(1) *Restrict* the set of coalitions or (2) consider $v(\cdot)$ with a specific *structure*

Cardinality-Restricted CFGs



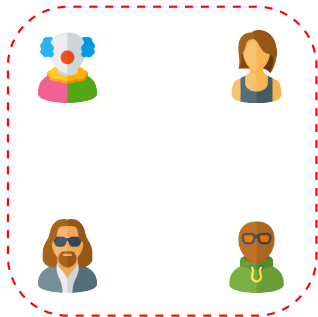
Maximum Cardinality k

E.g., only coalitions of size ≤ 3 are feasible

Polynomial Number of Coalitions

Total number of coalitions is $\sum_{i=1}^k \binom{|A|}{i} = \mathcal{O}(|A|^k)$, i.e., *polynomial* wrt $|A|$

Cardinality-Restricted CFGs



Maximum Cardinality k

E.g., only coalitions of size ≤ 3 are feasible

Polynomial Number of Coalitions

Total number of coalitions is $\sum_{i=1}^k \binom{|A|}{i} = \mathcal{O}(|A|^k)$, i.e., *polynomial* wrt $|A|$

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

- Combinatorial Auctions

- Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

- Ridesharing (Computational Challenge)

- Team Formation (Modelling Challenge)

Google Colab Hands-On Session

- Induced Subgraph Games

- Approximately Equivalent ISGs

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

- Combinatorial Auctions

- Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

- Ridesharing (Computational Challenge)

- Team Formation (Modelling Challenge)

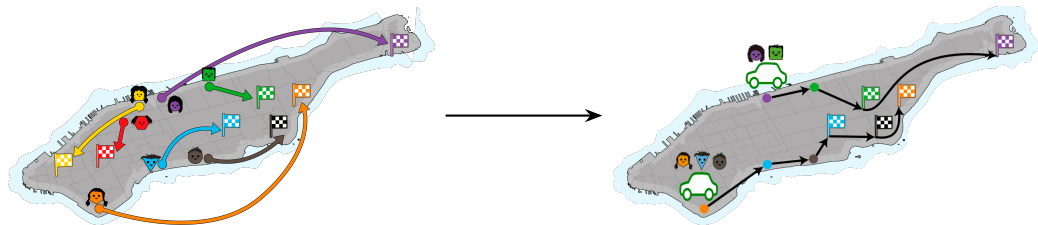
Google Colab Hands-On Session

- Induced Subgraph Games

- Approximately Equivalent ISGs

What is Ridesharing for Us?

Arrange *shared rides* (coalitions) among users that submit *real-time* requests, with the objective of *maximising* a given *utility measure* (e.g., cost / CO₂ reduction, etc.)



Ridesharing Solution Algorithm (Request Collection)

Incoming Requests

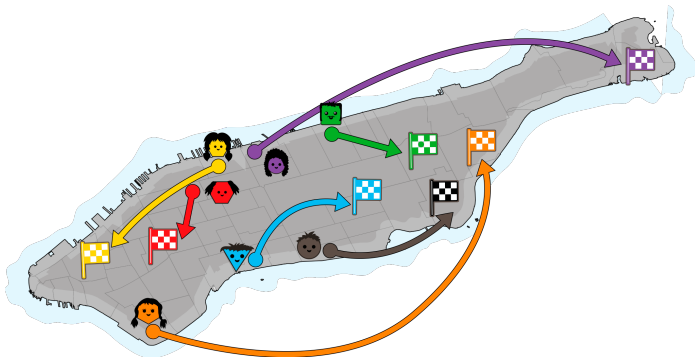


"I just issued a trip request"

Waiting Trip Requests



"I am waiting to share my ride"



Input of the Optimisation Problem

Example of a Ridesharing Request

“I want to go from point i to point j , and I am willing to wait δ minutes to be picked up by somebody ($d = \text{false}$) / before I leave with *my own car* ($d = \text{true}$)”

- $r = \langle i, j, d, \delta \rangle$ [A ridesharing request is a tuple r]
- $r \in R_t$ [The system receives a set R_t of requests at each time step t]
- $\langle R_1, \dots, R_t, \dots, R_h \rangle$ [Sequence of inputs over a time horizon h]
- The input sequence is *not known a priori* [Online optimisation problem]

Input of the Optimisation Problem

Example of a Ridesharing Request

“I want to go from point i to point j , and I am willing to wait δ minutes to be picked up by somebody ($d = \text{false}$) / before I leave with *my own car* ($d = \text{true}$)”

- $r = \langle i, j, d, \delta \rangle$ [A ridesharing request is a tuple r]
- $r \in R_t$ [The system receives a set R_t of requests at each time step t]
- $\langle R_1, \dots, R_t, \dots, R_h \rangle$ [Sequence of inputs over a time horizon h]
- The input sequence is *not known a priori* [Online optimisation problem]

Input of the Optimisation Problem

Example of a Ridesharing Request

“I want to go from point i to point j , and I am willing to wait δ minutes to be picked up by somebody ($d = \text{false}$) / before I leave with *my own car* ($d = \text{true}$)”

- $r = \langle i, j, d, \delta \rangle$ [A ridesharing request is a tuple r]
- $r \in R_t$ [The system receives a set R_t of requests at each time step t]
- $\langle R_1, \dots, R_t, \dots, R_h \rangle$ [Sequence of inputs over a time horizon h]
- The input sequence is *not known a priori* [Online optimisation problem]

Input of the Optimisation Problem

Example of a Ridesharing Request

“I want to go from point i to point j , and I am willing to wait δ minutes to be picked up by somebody ($d = \text{false}$) / before I leave with *my own car* ($d = \text{true}$)”

- $r = \langle i, j, d, \delta \rangle$ [A ridesharing request is a tuple r]
- $r \in R_t$ [The system receives a set R_t of requests at each time step t]
- $\langle R_1, \dots, R_t, \dots, R_h \rangle$ [Sequence of inputs over a time horizon h]
- The input sequence is *not known a priori* [Online optimisation problem]

Input of the Optimisation Problem

Example of a Ridesharing Request

“I want to go from point i to point j , and I am willing to wait δ minutes to be picked up by somebody ($d = \text{false}$) / before I leave with *my own car* ($d = \text{true}$)”

- $r = \langle i, j, d, \delta \rangle$ [A ridesharing request is a tuple r]
- $r \in R_t$ [The system receives a set R_t of requests at each time step t]
- $\langle R_1, \dots, R_t, \dots, R_h \rangle$ [Sequence of inputs over a time horizon h]
- The input sequence is *not known a priori* [Online optimisation problem]

Feasible Coalition S of Requests (Car)

- $|S| \leq k$ [Maximum cardinality constraint]
- $\min_{r_\alpha \in S} (t_\alpha + \delta_\alpha) \geq \max_{r_\beta \in S} t_\beta$ [Earliest req. in the pool when latest arrives]
- $\bigvee_{r_\gamma \in S} d_\gamma$ [At least one driver]
- ... [Some other constraints]

$$F(S) = |S| \leq k \wedge \min_{r_\alpha \in S} t_\alpha + \delta_\alpha \geq \max_{r_\beta \in S} t_\beta \wedge \bigvee_{r_\gamma \in S} d_\gamma \wedge \dots$$

- $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$ [Set of feasible coalitions from a set R of requests]

Feasible Coalition S of Requests (Car)

- $|S| \leq k$ [Maximum cardinality constraint]
- $\min_{r_\alpha \in S} (t_\alpha + \delta_\alpha) \geq \max_{r_\beta \in S} t_\beta$ [Earliest req. in the pool when latest arrives]
- $\bigvee_{r_\gamma \in S} d_\gamma$ [At least one driver]
- ... [Some other constraints]

$$F(S) = |S| \leq k \wedge \min_{r_\alpha \in S} t_\alpha + \delta_\alpha \geq \max_{r_\beta \in S} t_\beta \wedge \bigvee_{r_\gamma \in S} d_\gamma \wedge \dots$$

- $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$ [Set of feasible coalitions from a set R of requests]

Feasible Coalition S of Requests (Car)

- $|S| \leq k$ [Maximum cardinality constraint]
- $\min_{r_\alpha \in S} (t_\alpha + \delta_\alpha) \geq \max_{r_\beta \in S} t_\beta$ [Earliest req. in the pool when latest arrives]
- $\bigvee_{r_\gamma \in S} d_\gamma$ [At least one driver]
- ... [Some other constraints]

$$F(S) = |S| \leq k \wedge \min_{r_\alpha \in S} t_\alpha + \delta_\alpha \geq \max_{r_\beta \in S} t_\beta \wedge \bigvee_{r_\gamma \in S} d_\gamma \wedge \dots$$

- $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$ [Set of feasible coalitions from a set R of requests]

Feasible Coalition S of Requests (Car)

- $|S| \leq k$ [Maximum cardinality constraint]
- $\min_{r_\alpha \in S} (t_\alpha + \delta_\alpha) \geq \max_{r_\beta \in S} t_\beta$ [Earliest req. in the pool when latest arrives]
- $\bigvee_{r_\gamma \in S} d_\gamma$ [At least one driver]
- ... [Some other constraints]

$$F(S) = |S| \leq k \wedge \min_{r_\alpha \in S} t_\alpha + \delta_\alpha \geq \max_{r_\beta \in S} t_\beta \wedge \bigvee_{r_\gamma \in S} d_\gamma \wedge \dots$$

- $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$ [Set of feasible coalitions from a set R of requests]

Feasible Coalition S of Requests (Car)

- $|S| \leq k$ [Maximum cardinality constraint]
- $\min_{r_\alpha \in S} (t_\alpha + \delta_\alpha) \geq \max_{r_\beta \in S} t_\beta$ [Earliest req. in the pool when latest arrives]
- $\bigvee_{r_\gamma \in S} d_\gamma$ [At least one driver]
- ... [Some other constraints]

$$F(S) = |S| \leq k \wedge \min_{r_\alpha \in S} t_\alpha + \delta_\alpha \geq \max_{r_\beta \in S} t_\beta \wedge \bigvee_{r_\gamma \in S} d_\gamma \wedge \dots$$

- $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$ [Set of feasible coalitions from a set R of requests]

Feasible Coalition S of Requests (Car)

- $|S| \leq k$ [Maximum cardinality constraint]
- $\min_{r_\alpha \in S} (t_\alpha + \delta_\alpha) \geq \max_{r_\beta \in S} t_\beta$ [Earliest req. in the pool when latest arrives]
- $\bigvee_{r_\gamma \in S} d_\gamma$ [At least one driver]
- ... [Some other constraints]

$$F(S) = |S| \leq k \wedge \min_{r_\alpha \in S} t_\alpha + \delta_\alpha \geq \max_{r_\beta \in S} t_\beta \wedge \bigvee_{r_\gamma \in S} d_\gamma \wedge \dots$$

- $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$ [Set of feasible coalitions from a set R of requests]

Feasible Coalition S of Requests (Car)

- $|S| \leq k$ [Maximum cardinality constraint]
- $\min_{r_\alpha \in S} (t_\alpha + \delta_\alpha) \geq \max_{r_\beta \in S} t_\beta$ [Earliest req. in the pool when latest arrives]
- $\bigvee_{r_\gamma \in S} d_\gamma$ [At least one driver]
- ... [Some other constraints]

$$F(S) = |S| \leq k \wedge \min_{r_\alpha \in S} t_\alpha + \delta_\alpha \geq \max_{r_\beta \in S} t_\beta \wedge \bigvee_{r_\gamma \in S} d_\gamma \wedge \dots$$

- $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$ [Set of feasible coalitions from a set R of requests]

Value $V(S)$ of a Feasible Coalition S

- The *value* (utility) of a coalition S is defined as:

$$V(S) = \overbrace{\rho_{\text{CO}_2} \cdot E_{\text{CO}_2}(S) + \rho_{\text{noise}} \cdot E_{\text{noise}}(S) + \rho_{\text{traffic}} \cdot E_{\text{traffic}}(S)}^{\text{environmental benefits}} + \overbrace{\rho_{\text{QoS}} \cdot Q(S)}^{\text{quality of service}}$$

- $E_{\text{CO}_2}(S) = E_{\text{noise}}(S) = |S| \cdot \frac{\sum_{r \in S} d(\{r\}) - d(S)}{\sum_{r \in S} d(\{r\})}$ [Proportional to travelled distance]
- $E_{\text{traffic}}(S) = |S| - 1$ [Number of cars that have been avoided]
- $Q(S) = - \sum_{r \in S} \frac{\overbrace{t_r - t_r^*}^{\text{in-car delay}}}{t_r}$ [Relative difference with optimal arrival time t_r^*]

Value $V(S)$ of a Feasible Coalition S

- The *value* (utility) of a coalition S is defined as:

$$V(S) = \overbrace{\rho_{\text{CO}_2} \cdot E_{\text{CO}_2}(S) + \rho_{\text{noise}} \cdot E_{\text{noise}}(S) + \rho_{\text{traffic}} \cdot E_{\text{traffic}}(S)}^{\text{environmental benefits}} + \overbrace{\rho_{\text{QoS}} \cdot Q(S)}^{\text{quality of service}}$$

- $E_{\text{CO}_2}(S) = E_{\text{noise}}(S) = |S| \cdot \frac{\sum_{r \in S} d(\{r\}) - d(S)}{\sum_{r \in S} d(\{r\})}$ [Proportional to travelled distance]
- $E_{\text{traffic}}(S) = |S| - 1$ [Number of cars that have been avoided]
- $Q(S) = - \sum_{r \in S} \frac{\overbrace{t_r - t_r^*}^{\text{in-car delay}}}{t_r}$ [Relative difference with optimal arrival time t_r^*]

Value $V(S)$ of a Feasible Coalition S

- The *value* (utility) of a coalition S is defined as:

$$V(S) = \overbrace{\rho_{\text{CO}_2} \cdot E_{\text{CO}_2}(S) + \rho_{\text{noise}} \cdot E_{\text{noise}}(S) + \rho_{\text{traffic}} \cdot E_{\text{traffic}}(S)}^{\text{environmental benefits}} + \overbrace{\rho_{\text{QoS}} \cdot Q(S)}^{\text{quality of service}}$$

- $E_{\text{CO}_2}(S) = E_{\text{noise}}(S) = |S| \cdot \frac{\sum_{r \in S} d(\{r\}) - d(S)}{\sum_{r \in S} d(\{r\})}$ [Proportional to travelled distance]
- $E_{\text{traffic}}(S) = |S| - 1$ [Number of cars that have been avoided]
- $Q(S) = - \sum_{r \in S} \frac{\overbrace{t_r - t_r^*}^{\text{in-car delay}}}{t_r}$ [Relative difference with optimal arrival time t_r^*]

Value $V(S)$ of a Feasible Coalition S

- The *value* (utility) of a coalition S is defined as:

$$V(S) = \overbrace{\rho_{\text{CO}_2} \cdot E_{\text{CO}_2}(S) + \rho_{\text{noise}} \cdot E_{\text{noise}}(S) + \rho_{\text{traffic}} \cdot E_{\text{traffic}}(S)}^{\text{environmental benefits}} + \overbrace{\rho_{\text{QoS}} \cdot Q(S)}^{\text{quality of service}}$$

- $E_{\text{CO}_2}(S) = E_{\text{noise}}(S) = |S| \cdot \frac{\sum_{r \in S} d(\{r\}) - d(S)}{\sum_{r \in S} d(\{r\})}$ [Proportional to travelled distance]
- $E_{\text{traffic}}(S) = |S| - 1$ [Number of cars that have been avoided]
- $Q(S) = - \sum_{r \in S} \frac{\overbrace{t_r - t_r^*}^{\text{in-car delay}}}{t_r}$ [Relative difference with optimal arrival time t_r^*]

Optimal ILP Formulation

- Assume that $\langle R_1, \dots, R_t, \dots, R_h \rangle$ is *fully known a priori* [Offline problem]
- Let $R^U = \bigcup_{t=1}^h R_t$ [Set of all requests over the entire time horizon h]

Optimal ILP Formulation

$$\begin{aligned} &\text{maximise} && \sum_{S \in \mathcal{F}(R^U)} x_S \cdot V(S) \\ &\text{such that} && x_S + x_{S'} \leq 1 \quad \forall \mathcal{F}(R^U) : S \cap S' \neq \emptyset \end{aligned}$$

[Weighted set packing]

Optimal ILP Formulation

- Assume that $\langle R_1, \dots, R_t, \dots, R_h \rangle$ is *fully known a priori* [Offline problem]
- Let $R^U = \bigcup_{t=1}^h R_t$ [Set of all requests over the entire time horizon h]

Optimal ILP Formulation

$$\begin{aligned} &\text{maximise} && \sum_{S \in \mathcal{F}(R^U)} x_S \cdot V(S) \\ &\text{such that} && x_S + x_{S'} \leq 1 \quad \forall \mathcal{F}(R^U) : S \cap S' \neq \emptyset \end{aligned}$$

[Weighted set packing]

Optimal ILP Formulation

- Assume that $\langle R_1, \dots, R_t, \dots, R_h \rangle$ is *fully known a priori* [Offline problem]
- Let $R^U = \bigcup_{t=1}^h R_t$ [Set of all requests over the entire time horizon h]

Optimal ILP Formulation

$$\text{maximise} \quad \sum_{S \in \mathcal{F}(R^U)} x_S \cdot V(S)$$

[Weighted set packing]

$$\text{such that} \quad x_S + x_{S'} \leq 1 \quad \forall \mathcal{F}(R^U) : S \cap S' \neq \emptyset$$

Optimal ILP Formulation

- Assume that $\langle R_1, \dots, R_t, \dots, R_h \rangle$ is *fully known a priori* [Offline problem]
- Let $R^\cup = \bigcup_{t=1}^h R_t$ [Set of all requests over the entire time horizon h]

Optimal ILP Formulation

$$\begin{aligned} &\text{maximise} && \sum_{S \in \mathcal{F}(R^\cup)} x_S \cdot V(S) \\ &\text{such that} && x_S + x_{S'} \leq 1 \quad \forall \mathcal{F}(R^\cup) : S \cap S' \neq \emptyset \end{aligned}$$

[Weighted set packing]

Curse of Dimensionality

- Recall that $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$
- With $|S| \leq k$, $|\mathcal{F}(R)| \leq \sum_{i=1}^k \binom{|R|}{i}$, i.e., $\mathcal{O}(|R|^k)$ [Polynomial complexity]
- In practice, $|R_t|$ can be as high as 400 [Request rate in NY taxi dataset]

Scalability Problem

Enumerating all coalitions in $\mathcal{F}(R)$ is impractical, especially in realistic application scenarios with *very limited time budget* for the solution

Our Solution

Consider a restricted set $\hat{\mathcal{F}}(R)$ of *good candidate coalitions* instead of $\mathcal{F}(R)$

Curse of Dimensionality

- Recall that $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$
- With $|S| \leq k$, $|\mathcal{F}(R)| \leq \sum_{i=1}^k \binom{|R|}{i}$, i.e., $\mathcal{O}(|R|^k)$ [Polynomial complexity]
- In practice, $|R_t|$ can be as high as 400 [Request rate in NY taxi dataset]

Scalability Problem

Enumerating all coalitions in $\mathcal{F}(R)$ is impractical, especially in realistic application scenarios with *very limited time budget* for the solution

Our Solution

Consider a restricted set $\hat{\mathcal{F}}(R)$ of *good candidate coalitions* instead of $\mathcal{F}(R)$

Curse of Dimensionality

- Recall that $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$
- With $|S| \leq k$, $|\mathcal{F}(R)| \leq \sum_{i=1}^k \binom{|R|}{i}$, i.e., $\mathcal{O}(|R|^k)$ [Polynomial complexity]
- In practice, $|R_t|$ can be as high as 400 [Request rate in NY taxi dataset]

Scalability Problem

Enumerating all coalitions in $\mathcal{F}(R)$ is impractical, especially in realistic application scenarios with *very limited time budget* for the solution

Our Solution

Consider a restricted set $\hat{\mathcal{F}}(R)$ of *good candidate coalitions* instead of $\mathcal{F}(R)$

Curse of Dimensionality

- Recall that $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$
- With $|S| \leq k$, $|\mathcal{F}(R)| \leq \sum_{i=1}^k \binom{|R|}{i}$, i.e., $\mathcal{O}(|R|^k)$ [Polynomial complexity]
- In practice, $|R_t|$ can be as high as 400 [Request rate in NY taxi dataset]

Scalability Problem

Enumerating all coalitions in $\mathcal{F}(R)$ is impractical, especially in realistic application scenarios with *very limited time budget* for the solution

Our Solution

Consider a restricted set $\hat{\mathcal{F}}(R)$ of *good candidate coalitions* instead of $\mathcal{F}(R)$

Curse of Dimensionality

- Recall that $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$
- With $|S| \leq k$, $|\mathcal{F}(R)| \leq \sum_{i=1}^k \binom{|R|}{i}$, i.e., $\mathcal{O}(|R|^k)$ [Polynomial complexity]
- In practice, $|R_t|$ can be as high as 400 [Request rate in NY taxi dataset]

Scalability Problem

Enumerating all coalitions in $\mathcal{F}(R)$ is impractical, especially in realistic application scenarios with *very limited time budget* for the solution

Our Solution

Consider a restricted set $\hat{\mathcal{F}}(R)$ of *good candidate coalitions* instead of $\mathcal{F}(R)$

Curse of Dimensionality

- Recall that $\mathcal{F}(R) = \{S \in 2^R \mid F(S)\}$
- With $|S| \leq k$, $|\mathcal{F}(R)| \leq \sum_{i=1}^k \binom{|R|}{i}$, i.e., $\mathcal{O}(|R|^k)$ [Polynomial complexity]
- In practice, $|R_t|$ can be as high as 400 [Request rate in NY taxi dataset]

Scalability Problem

Enumerating all coalitions in $\mathcal{F}(R)$ is impractical, especially in realistic application scenarios with *very limited time budget* for the solution

Our Solution

Consider a restricted set $\hat{\mathcal{F}}(R)$ of *good candidate coalitions* instead of $\mathcal{F}(R)$

Ridesharing Solution Algorithm (Candidate Generation)

- ☁ CO₂ emissions
- 📢 Acoustic pollution
- 🚗 Traffic congestion
- 🕒 Quality of service



20 seconds



Probabilistic
Greedy
Algorithm

Candidate
Cars

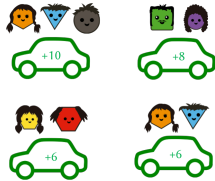


...



Ridesharing Solution Algorithm (ILP Optimisation)

Good Candidates



40 seconds



Integer Linear
Programming
Solver

ILP
Solution



Approximated ILP Formulation

$$\text{maximise} \quad \sum_{S \in \hat{\mathcal{F}}(R^U)} x_S \cdot V(S)$$

[Only good candidates]

$$\text{such that} \quad x_S + x_{S'} \leq 1 \quad \forall \hat{\mathcal{F}}(R^U) : S \cap S' \neq \emptyset$$

Computational Advantage

Approximated ILP has a number of variables that is $< 0.01\%$ of the optimal ILP

Quality of Approximated Solutions

Approximated solutions have a quality that is $> 95\%$ of the optimal one

Approximated ILP Formulation

$$\text{maximise} \quad \sum_{S \in \hat{\mathcal{F}}(R^U)} x_S \cdot V(S)$$

[Only good candidates]

$$\text{such that} \quad x_S + x_{S'} \leq 1 \quad \forall \hat{\mathcal{F}}(R^U) : S \cap S' \neq \emptyset$$

Computational Advantage

Approximated ILP has a number of variables that is $< 0.01\%$ of the optimal ILP

Quality of Approximated Solutions

Approximated solutions have a quality that is $> 95\%$ of the optimal one

Approximated ILP Formulation

$$\text{maximise} \quad \sum_{S \in \hat{\mathcal{F}}(R^U)} x_S \cdot V(S)$$

[Only good candidates]

$$\text{such that} \quad x_S + x_{S'} \leq 1 \quad \forall \hat{\mathcal{F}}(R^U) : S \cap S' \neq \emptyset$$

Computational Advantage

Approximated ILP has a number of variables that is $< 0.01\%$ of the optimal ILP

Quality of Approximated Solutions

Approximated solutions have a quality that is $> 95\%$ of the optimal one

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

Combinatorial Auctions

Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

Ridesharing (Computational Challenge)

Team Formation (Modelling Challenge)

Google Colab Hands-On Session

Induced Subgraph Games

Approximately Equivalent ISGs

What is Team Formation for Us?

Partition a classroom A into *proficient* and *congenial* teams of size k [$|A| = m \cdot k$]



Input of the Optimisation Problem

Student Representation

- $g \in \{\text{man}, \text{woman}\}$ stands for the student's gender
 - $p \in [-1, 1]^4$ is a *personality profile* with 4 personality traits
 - $l : C \rightarrow [0, 1]$ gives the student's level $l(c)$ for competence $c \in C$
- } [Congeniality]

Task Representation (Same for all Teams)

A task τ requires a *competence level* met by at least one student [Proficiency]

Input of the Optimisation Problem

Student Representation

- $g \in \{\text{man}, \text{woman}\}$ stands for the student's gender
 - $p \in [-1, 1]^4$ is a *personality profile* with 4 personality traits
 - $l : C \rightarrow [0, 1]$ gives the student's level $l(c)$ for competence $c \in C$
- } [Congeniality]

Task Representation (Same for all Teams)

A task τ requires a *competence level* met by at least one student [Proficiency]

Input of the Optimisation Problem

Student Representation

- $g \in \{\text{man}, \text{woman}\}$ stands for the student's gender
 - $p \in [-1, 1]^4$ is a *personality profile* with 4 personality traits
 - $l : C \rightarrow [0, 1]$ gives the student's level $l(c)$ for competence $c \in C$
- } [Congeniality]

Task Representation (Same for all Teams)

A task τ requires a *competence level* met by at least one student [Proficiency]

Value $V(S)$ of a Team S given a Task τ

- The *value* (utility) of a team $S \in [A]^k$ given a task τ is defined as:

$$V(S, \tau) = \lambda \cdot \overbrace{U_{\text{prof}}(S, \tau)}^{\text{proficiency}} + (1 - \lambda) \cdot \overbrace{U_{\text{cong}}(S)}^{\text{congeniality}} \quad [\lambda = \text{proficiency importance}]$$

- Given a partition \mathcal{S} of A into teams of size k , the value of \mathcal{S} is defined as:

$$V(\mathcal{S}, \tau) = \prod_{S \in \mathcal{S}} V(S, \tau) \quad [\text{Bernoulli-Nash product}]$$

Value $V(S)$ of a Team S given a Task τ

- The *value* (utility) of a team $S \in [A]^k$ given a task τ is defined as:

$$V(S, \tau) = \lambda \cdot \overbrace{U_{\text{prof}}(S, \tau)}^{\text{proficiency}} + (1 - \lambda) \cdot \overbrace{U_{\text{cong}}(S)}^{\text{congeniality}} \quad [\lambda = \text{proficiency importance}]$$

- Given a partition \mathcal{S} of A into teams of size k , the value of \mathcal{S} is defined as:

$$V(\mathcal{S}, \tau) = \prod_{S \in \mathcal{S}} V(S, \tau) \quad [\text{Bernoulli-Nash product}]$$

Non-linear IP Formulation

$$\begin{array}{ll}\text{maximise} & \prod_{S \in [A]^k} V(S, \tau)^{x_S} \quad [V(S, \tau)^{x_S} = V(S, \tau) \text{ if } x_S = 1, 1 \text{ otherwise}] \\ \text{subject to} & \sum_{S \in [A]^k} x_S = m \quad [\text{Partition of exactly } m \text{ teams}] \\ & \sum_{S \in [A]^k} \underbrace{M_{iS}}_{i \in S} \cdot x_S = 1 \quad \forall i \in A \quad [\text{No overlapping teams}]\end{array}$$

Modelling Problem

$\prod_{S \in [A]^k} V(S, \tau)^{x_S}$ is *not* a linear function, cannot be solved with ILP solvers

Non-linear IP Formulation

$$\begin{array}{ll}\text{maximise} & \prod_{S \in [A]^k} V(S, \tau)^{x_S} \quad [V(S, \tau)^{x_S} = V(S, \tau) \text{ if } x_S = 1, 1 \text{ otherwise}] \\ \text{subject to} & \sum_{S \in [A]^k} x_S = m \quad [\text{Partition of exactly } m \text{ teams}] \\ & \sum_{S \in [A]^k} \underbrace{M_{iS}}_{i \in S} \cdot x_S = 1 \quad \forall i \in A \quad [\text{No overlapping teams}]\end{array}$$

Modelling Problem

$\prod_{S \in [A]^k} V(S, \tau)^{x_S}$ is *not* a linear function, cannot be solved with ILP solvers

Positive Monotonic Functions

Applying a *positive monotonic* (PM) function to the objective does *not* change the optimum, since the order among solutions is preserved

Question

Which PM function $g(\cdot)$ should I pick such that $g\left(\prod_{S \in [A]^k} V(S, \tau)^{x_S}\right)$ is linear?

Solution

- \log is a PM function in the considered domain
- $\log\left(\prod_{S \in [A]^k} V(S, \tau)^{x_S}\right) = \sum_{S \in [A]^k} x_S \cdot \underbrace{\log(V(S, \tau))}_{\text{constant value}}$ [Linear objective function]

Positive Monotonic Functions

Applying a *positive monotonic* (PM) function to the objective does *not* change the optimum, since the order among solutions is preserved

Question

Which PM function $g(\cdot)$ should I pick such that $g\left(\prod_{S \in [A]^k} V(S, \tau)^{x_S}\right)$ is linear?

Solution

- \log is a PM function in the considered domain
- $\log\left(\prod_{S \in [A]^k} V(S, \tau)^{x_S}\right) = \sum_{S \in [A]^k} x_S \cdot \underbrace{\log(V(S, \tau))}_{\text{constant value}}$ [Linear objective function]

Positive Monotonic Functions

Applying a *positive monotonic* (PM) function to the objective does *not* change the optimum, since the order among solutions is preserved

Question

Which PM function $g(\cdot)$ should I pick such that $g\left(\prod_{S \in [A]^k} V(S, \tau)^{x_S}\right)$ is linear?

Solution

- \log is a PM function in the considered domain
- $\log\left(\prod_{S \in [A]^k} V(S, \tau)^{x_S}\right) = \sum_{S \in [A]^k} x_S \cdot \underbrace{\log(V(S, \tau))}_{\text{constant value}}$ [Linear objective function]

Linearised ILP Formulation

$$\begin{array}{ll}\text{maximise} & \sum_{S \in [A]^k} x_S \cdot \log(V(S, \tau)) \\ \text{subject to} & \sum_{S \in [A]^k} x_S = m \\ & \sum_{S \in [A]^k} M_{iS} \cdot x_S = 1 \quad \forall i \in A\end{array}$$

Further Reading

Andrejczuk *et al.*, “Synergistic Team Composition: A Computational Approach to Foster Diversity in Teams”, *Knowledge-Based Systems*, 2019

Linearised ILP Formulation

$$\begin{array}{ll}\text{maximise} & \sum_{S \in [A]^k} x_S \cdot \log(V(S, \tau)) \\ \text{subject to} & \sum_{S \in [A]^k} x_S = m \\ & \sum_{S \in [A]^k} M_{iS} \cdot x_S = 1 \quad \forall i \in A\end{array}$$

Further Reading

Andrejczuk *et al.*, “Synergistic Team Composition: A Computational Approach to Foster Diversity in Teams”, *Knowledge-Based Systems*, 2019

Further Reading

- Boyd and Vandenberghe, *Convex Optimization*, 2004
- Hentenryck and Bent, *Online Stochastic Combinatorial Optimization*, 2009
- Bistaffa *et al.*, “A Computational Approach to Quantify the Benefits of Ridesharing for Policy Makers and Travellers”, *IEEE Transactions on Intelligent Transportation Systems*, 2021
- Andrejczuk *et al.*, “Synergistic Team Composition: A Computational Approach to Foster Diversity in Teams”, *Knowledge-Based Systems*, 2019

See you tomorrow!

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

- Combinatorial Auctions

- Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

- Ridesharing (Computational Challenge)

- Team Formation (Modelling Challenge)

Google Colab Hands-On Session

- Induced Subgraph Games

- Approximately Equivalent ISGs

Google Colab Hands-On Session

1. Weighted Knapsack Problem
<https://bit.ly/aihub2022-wk>
2. Weighted Set Packing Problem
<https://bit.ly/aihub2022-wsp>
3. Coalition Structure Generation
<https://bit.ly/aihub2022-csg>
4. Approximately Equivalent ISG
<https://bit.ly/aihub2022-aeisg>
5. CSG on ISGs as Graph Clustering
<https://bit.ly/aihub2022-gc>

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

- Combinatorial Auctions

- Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

- Ridesharing (Computational Challenge)

- Team Formation (Modelling Challenge)

Google Colab Hands-On Session

- Induced Subgraph Games

- Approximately Equivalent ISGs

Characteristic Function

Characteristic Function

The function $v : \mathcal{P}(A) \rightarrow \mathbb{R}$ associates a value to *every coalition* (i.e., subset) of A

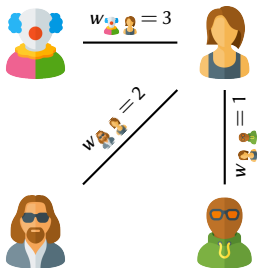
Exponential Complexity

Representing $v(\cdot)$ as a *table* requires an *exponential* number of steps (i.e., $2^{|A|}$)

Mitigate this Complexity

(1) *Restrict* the set of coalitions or (2) consider $v(\cdot)$ with a specific *structure*

Induced Subgraph Games (ISGs)



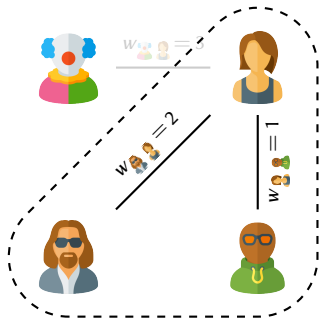
Weighted Graph G among Agents

$$G_w = (\{\text{robot head, woman, man with sunglasses, man with glasses}\}, \underbrace{\{(\text{robot head, woman}), (\text{robot head, man with sunglasses})\}}_2, \underbrace{\{(\text{robot head, man with glasses}), (\text{woman, man with glasses})\}}_3, \underbrace{\{(\text{man with sunglasses, man with glasses})\}}_1)$$

Value is the Sum of Induced Edges

$$v(\{\text{man with sunglasses, woman, man with glasses}\}) = 2 + 1 = 3$$

Induced Subgraph Games (ISGs)



Weighted Graph G among Agents

$$G_w = (\{\text{robot}, \text{woman}, \text{man with sunglasses}, \text{man with glasses}\}, \underbrace{\{(\text{robot}, \text{woman})\}}_2, \underbrace{\{(\text{robot}, \text{man with sunglasses}), (\text{robot}, \text{man with glasses})\}}_3, \underbrace{\{(\text{man with sunglasses}, \text{man with glasses})\}}_1)$$

Value is the Sum of Induced Edges

$$v(\{\text{man with sunglasses}, \text{man with glasses}\}) = 2 + 1 = 3$$

Induced Subgraph Games (ISGs)

Succinct Game Representation

The characteristic function is *entirely* represented by the weighted graph G_w

Computational Advantages

CSG on ISGs can be treated as a *graph clustering* problem (“easier” than CSG)

Limited Representation Power

Not every characteristic function game can be *perfectly* represented as an ISG

Induced Subgraph Games (ISGs)

Succinct Game Representation

The characteristic function is *entirely* represented by the weighted graph G_w

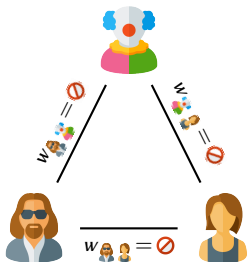
Computational Advantages

CSG on ISGs can be treated as a *graph clustering* problem (“easier” than CSG)

Limited Representation Power

Not every characteristic function game can be *perfectly* represented as an ISG

ISGs Cannot Represent Every CFG



$$v(S) = \begin{cases} 0, & \text{if } |S| = 1, \\ 1, & \text{if } |S| = 2, \\ 6, & \text{if } |S| = 3. \end{cases}$$

Computational Sustainability in Multi-Agent Systems

Theoretical Foundations of Constrained Optimisation in MAS

Combinatorial Auctions

Characteristic Function Games

Practical Applications of Constrained Optimisation in MAS

Ridesharing (Computational Challenge)

Team Formation (Modelling Challenge)

Google Colab Hands-On Session

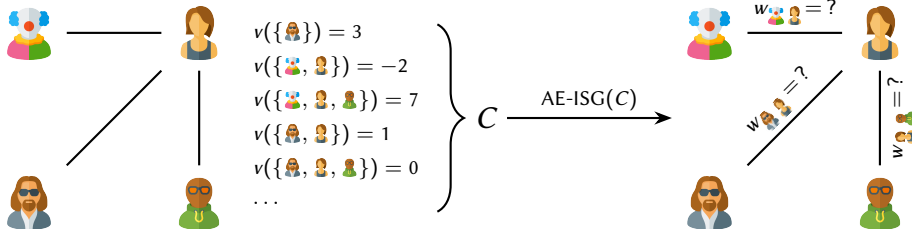
Induced Subgraph Games

Approximately Equivalent ISGs

Can We Approximate a CFG as an ISG?

Approximately Equivalent ISG (AE-ISG)

Given a CFG C , compute the ISG that *best approximates* C , namely $\text{AE-ISG}(C)$



AE-ISG as Norm Approximation (ℓ_p Linear Regression)

minimise $\| \underbrace{Mw - v}_{\text{residuals}} \|_p$

$$M = \begin{matrix} \begin{matrix} \text{👤👤} \\ \text{👤👤} \end{matrix} & \begin{matrix} \text{👤👤} \\ \text{👤👤} \end{matrix} & \begin{matrix} \text{👤👤} \\ \text{👤👤} \end{matrix} \\ \text{w} & \text{w} & \text{w} \\ \left[\begin{array}{ccc} \vdots & \vdots & \vdots \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ \vdots & \vdots & \vdots \end{array} \right] \end{matrix}$$

$$v = \left[\begin{array}{c} \vdots \\ v(\{\text{👤👤}, \text{👤}, \text{👤}\}) \\ v(\{\text{👤👤}, \text{👤}\}) \\ v(\{\text{👤👤}, \text{👤}, \text{👤}\}) \\ \vdots \end{array} \right]$$

AE-ISG as Norm Approximation (ℓ_p Linear Regression)

$$\text{minimise } \underbrace{\|Mw - v\|}_\text{residuals}_p$$

Residual Vector

The *residual vector* $r = Mw - v$ is the vector of *differences* between *approximated* coalitional values (i.e., Mw) and *original* coalitional values (i.e., v)

Constrained Norm Approximation

Some coalitions (singletons) can be represented *exactly* via additional constraints

AE-ISG as Norm Approximation (ℓ_p Linear Regression)

Size of AE-ISG Model

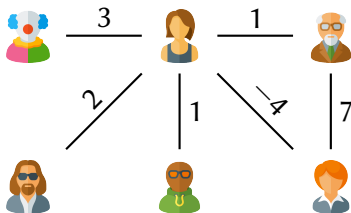
Building M and v requires to go through the set of coalitional values (obviously)

Computational Complexity

If the set of feasible coalitions is *polynomial* (e.g., ridesharing), computing $\text{AE-ISG}(C)$ has a *manageable* complexity, depending on the norm ℓ_p :

- $\ell_1/\ell_\infty \rightarrow$ Linear Programming (exact, CPU)
- $\ell_2 \rightarrow$ Least Squares (exact/analytical, GPU)
- $\ell_{>2} \rightarrow$ Iteratively Reweighted Least Squares (numerical)

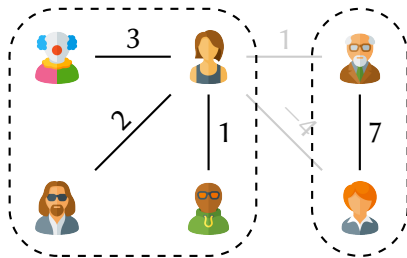
CSG on ISGs as Graph Clustering



CSG on ISGs Optimisation Objective

Maximise sum of of *clusters' internal weights* (namely, *coverage* measure)

CSG on ISGs as Graph Clustering



CSG on ISGs Optimisation Objective

Maximise sum of of *clusters' internal weights* (namely, *coverage* measure)

ILP for Optimal Graph Clustering

$X_{ij} = 1 \rightarrow$ edge $\{i, j\}$ is “activated” (i and j are in the same cluster)

maximise $\sum_{i,j \in A} w_{ij} \cdot X_{ij}$ [Coverage objective function]

subject to $\forall i, j, z \in A : \begin{cases} X_{ij} + X_{jz} - 2 \cdot X_{iz} \leq 1 \\ X_{iz} + X_{ij} - 2 \cdot X_{jz} \leq 1 \\ X_{jz} + X_{iz} - 2 \cdot X_{ij} \leq 1 \end{cases}$ [Transitivity]

$\forall i \in A : \sum_{j \in A} X_{ij} \leq k$ [Cardinality constraint]

Google Colab Hands-On Session

1. Weighted Knapsack Problem
<https://bit.ly/aihub2022-wk>
2. Weighted Set Packing Problem
<https://bit.ly/aihub2022-wsp>
3. Coalition Structure Generation
<https://bit.ly/aihub2022-csg>
4. Approximately Equivalent ISG
<https://bit.ly/aihub2022-aeisg>
5. CSG on ISGs as Graph Clustering
<https://bit.ly/aihub2022-gc>