Stabilizing Grand Cooperation of Machine Scheduling Game via Setup Cost Pricing

Lindong Liu

School of Management; International Institute of Finance

University of Science and Technology of China

Co-authored with Zikang Li (PG Student, USTC)

NWPU-Online, July, 2020

Outline

Preliminaries

Cooperative Game

A **cooperative game** is defined by a pair (V, C):

- A set $V = \{1, 2, \ldots, v\}$ of players, **grand colaition**;
- A **characteristic function** $C(S) =$ the minimum total cost achieved by the cooperation of members in coalition $S \in \mathbb{S} = 2^V \setminus \{\emptyset\}$.

The game requires:

- A **cost allocation** $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_v] \in \mathbb{R}^v$, where $\alpha_k =$ the cost allocated to each player $k \in V$.

Core

Define $\alpha(S) = \sum_{k \in S} \alpha_k$.

A cost allocation $\alpha \in \mathbb{R}^v$ is in the **core** if it satisfies:

- **Budget Balance** Constraint: $\alpha(V) = C(V)$;
- **Coalition Stability** Constraints: $\alpha(S) \leq C(S)$ for each $S \in \mathbb{S}$.

$$\mathrm{Core}(V, C) = \left\{ \alpha : \ \alpha(V) = C(V), \right.$$
$$\left. \alpha(S) \leq C(S), \ \forall S \in \mathbb{S} \setminus \{V\}, \ \alpha \in \mathbb{R}^v \right\}.$$

Core

Define $\alpha(S) = \sum_{k \in S} \alpha_k$.

A cost allocation $\alpha \in \mathbb{R}^v$ is in the **core** if it satisfies:

- **Budget Balance** Constraint: $\alpha(V) = C(V)$;
- **Coalition Stability** Constraints: $\alpha(S) \leq C(S)$ for each $S \in \mathbb{S}$.

$$\mathrm{Core}(V, C) = \Big\{\alpha : \ \alpha(V) = C(V), \\ \alpha(S) \leq C(S), \ \forall S \in \mathbb{S} \setminus \{V\}, \ \alpha \in \mathbb{R}^v\Big\}.$$

However, $\mathrm{Core}(V, c)$ can be empty.

Existing Instruments

$$\mathrm{Core}(V, C) = \left\{ \alpha : \ \alpha(V) = C(V), \ \alpha(S) \leq C(S), \ \forall S \in \mathbb{S} \setminus \{V\} \right\}$$

Existing Instruments

$$\mathrm{Core}(V, C) = \left\{ \alpha : \ \alpha(V) = C(V), \ \alpha(S) \leq C(S), \ \forall S \in \mathbb{S} \setminus \{V\} \right\}$$

- Subsidization: $\alpha(V) = C(V) - \theta$, $\epsilon-$**core**;

Existing Instruments

$$\mathrm{Core}(V, C) = \left\{\alpha : \ \alpha(V) = C(V), \ \alpha(S) \leq C(S), \ \forall S \in \mathbb{S} \setminus \{V\}\right\}$$

- Subsidization: $\alpha(V) = C(V) - \theta$, $\epsilon-$**core**;
- Penalization: $\alpha(S) \leq C(S) + z$, **least core**;

Existing Instruments

$$\mathrm{Core}(V, C) = \left\{ \alpha : \ \alpha(V) = C(V), \ \alpha(S) \le C(S), \ \forall S \in \mathbb{S} \setminus \{V\} \right\}$$

- Subsidization: $\alpha(V) = C(V) - \theta$, $\epsilon-$**core**;
- Penalization: $\alpha(S) \le C(S) + z$, **least core**;
- Simul. S & P: $\alpha(V) = C(V) - \theta$ and $\alpha(S) \le C(S) + z$, **PSF**;

Existing Instruments

$$\mathrm{Core}(V, C) = \left\{ \alpha : \ \alpha(V) = C(V), \ \alpha(S) \leq C(S), \ \forall S \in \mathbb{S} \setminus \{V\} \right\}$$

- Subsidization: $\alpha(V) = C(V) - \theta$, $\epsilon-$**core**;
- Penalization: $\alpha(S) \leq C(S) + z$, **least core**;
- Simul. S & P: $\alpha(V) = C(V) - \theta$ and $\alpha(S) \leq C(S) + z$, **PSF**;
- Inv. Opt.: Changing c to d such that $\mathrm{Core}(V, D)$ is non-empty.

Existing Instruments

$$\mathrm{Core}(V, C) = \left\{ \alpha : \ \alpha(V) = C(V), \ \alpha(S) \leq C(S), \ \forall S \in \mathbb{S} \setminus \{V\} \right\}$$

- Subsidization: $\alpha(V) = C(V) - \theta$, $\epsilon-$**core**;
- Penalization: $\alpha(S) \leq C(S) + z$, **least core**;
- Simul. S & P: $\alpha(V) = C(V) - \theta$ and $\alpha(S) \leq C(S) + z$, **PSF**;
- Inv. Opt.: Changing c to d such that $\mathrm{Core}(V, D)$ is non-empty.

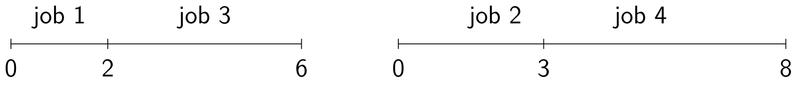
S.	Caprara and Letchford (2010, MP), Liu et al. (2016, IJOC)
P.	Faigle et al. (2001, IJGT), Schulz and Uhan (2010, OR)
P&S	Liu et al. (2018, OR)
Inv. Opt.	Liu et al. (2020, under review)

Illustrative Example

Example: Machine Scheduling Game (MSG)

Game of Parallel Machine Scheduling with Setup Cost:

- Grand coalition: $V = \{1, 2, 3, 4\}$;
- Processing times: $t_1 = 2$, $t_2 = 3$, $t_3 = 4$, $t_4 = 5$;
- Machine setup cost: $t_0 = 9.5$;
- $c(S)$ for $S \in \mathbb{S}$: minimizes the total completion time of jobs in S plus the machine setup cost;
- $\pi(N) = \pi(\{1,3\}) + \pi(\{2,4\}) = 38$ (SPT Rule).



Example: Empty Core

Coalitions	Cost
$\{1\}$	11.5
$\{2\}$	12.5
$\{3\}$	13.5
$\{4\}$	14.5
$\{1,2\}$	16.5
$\{1,3\}$	17.5
$\{1,4\}$	18.5
$\{2,3\}$	19.5
$\{2,4\}$	20.5
$\{3,4\}$	22.5
$\{1,2,3\}$	25.5
$\{1,2,4\}$	26.5
$\{1,3,4\}$	28.5
$\{2,3,4\}$	31.5
$\{1,2,3,4\}$	38

Example: Empty Core

Coalitions	Cost
$\{1\}$	11.5
$\{2\}$	12.5
$\{3\}$	13.5
$\{4\}$	14.5
$\{1,2\}$	16.5
$\{1,3\}$	17.5
$\{1,4\}$	18.5
$\{2,3\}$	19.5
$\{2,4\}$	20.5
$\{3,4\}$	22.5
$\{1,2,3\}$	25.5
$\{1,2,4\}$	26.5
$\{1,3,4\}$	28.5
$\{2,3,4\}$	31.5
$\{1,2,3,4\}$	38

Optimal Cost Allocation Problem

$$\max \ \left(\alpha_1+\alpha_2+\alpha_3+\alpha_4\right) = 37.25 < 38$$
$$s.t. \ \ \alpha_1 \le 11.5, \ \cdots, \ \alpha_4 \le 14.5,$$
$$\alpha_1+\alpha_2 \le 16.5, \ \cdots, \ \alpha_3+\alpha_4 \le 22.5,$$
$$\cdots,$$
$$\alpha_1+\alpha_2+\alpha_3+\alpha_4 \le 38.$$

$$\alpha^* = \left[6; 8.75; 10.75; 11.75\right]$$

Models & Analyses

Problem Definition and Formulation

Definition

An Identical Variable Parallel machine scheduling of Unweighted jobs game (IVPU):

Problem Definition and Formulation

Definition

An Identical Variable Parallel machine scheduling of Unweighted jobs game (IVPU):

- Grand coalition: $V = \{1, 2, \ldots, v\}$;

Problem Definition and Formulation

Definition

An Identical Variable Parallel machine scheduling of Unweighted jobs game (IVPU):

- Grand coalition: $V = \{1, 2, \ldots, v\}$;
- Identical machine: $M = \{1, 2, \ldots, m\}$;

Problem Definition and Formulation

Definition

An Identical Variable Parallel machine scheduling of Unweighted jobs game (IVPU):

- Grand coalition: $V = \{1, 2, \ldots, v\}$;
- Identical machine: $M = \{1, 2, \ldots, m\}$;
- Each machine Price: P and Each job processing time: t_k;

Problem Definition and Formulation

Definition

An Identical Variable Parallel machine scheduling of Unweighted jobs game (IVPU):

- Grand coalition: $V = \{1, 2, \ldots, v\}$;
- Identical machine: $M = \{1, 2, \ldots, m\}$;
- Each machine Price: P and Each job processing time: t_k;
- Characteristic function: $c(S) = \min(\sum_{k \in S} C_k + Pm_S)$,

  where C_k is the completion time of job $k \in S$ and m_S is the number of using machine for the sub-coalition S.

Problem Definition and Formulation

Definition

A cooperative TU game (V,c) is called an IVPU game if it satisfies the following formulations:

Problem Definition and Formulation

Definition

A cooperative TU game (V,c) is called an IVPU game if it satisfies the following formulations:

$$
\begin{aligned}
c(S,P) = &\min \sum_{k\in V}\sum_{j\in O} c_{kj}x_{kj} + P\sum_{k\in s} x_{k1} \\
s.t. \quad & \sum_{j\in O} x_{kj} - y_k^s = 0, \forall k \in V, \\
& \sum_{k\in V} x_{kj} \leq m, \forall j \in O, \\
& x_{kj} \in \{0,1\}, \forall k \in V, \forall j \in O, \\
& y_k^s = 1, k \in s\,; y_k^s = 0, k \notin s.
\end{aligned}
$$

Properties

Definition

- The interval $[P_L(m, S), P_H(m, S)]$ denotes the value range of price when the number of using machines m and the coalition S are given.

Properties

Definition

- The interval $[P_L(m, S), P_H(m, S)]$ denotes the value range of price when the number of using machines m and the coalition S are given.
- For the grand coalition V, define $P_L(v, V) = 0, P_H(1, V) = P^*$. So the practical domain of the price is $[0, P^*]$ which is divided into v nonoverlapping subintervals by the number of using machines.

Properties

Definition

- The interval $[P_L(m,S), P_H(m,S)]$ denotes the value range of price when the number of using machines m and the coalition S are given.
- For the grand coalition V, define $P_L(v,V) = 0, P_H(1,V) = P^*$. So the practical domain of the price is $[0, P^*]$ which is divided into v nonoverlapping subintervals by the number of using machines.
- Denote the right end of every subinterval as $P_i, 1 \leq i \leq v$, where $P_1 = P^*$.

Properties

$$\begin{aligned}\omega(P) \quad &= \min_\alpha\{c(V, m(V, P)) - \alpha(V) : \\ &\alpha(s) \le c(s, m(s, P)), \forall s \in S, \alpha \in \mathbb{R}^v\};\end{aligned}$$

Properties

$$\begin{aligned}\omega(P) \quad &= \min_\alpha\{c(V, m(V,P)) - \alpha(V) : \\ &\alpha(s) \le c(s, m(s,P)), \forall s \in S, \alpha \in \mathbb{R}^v\};\end{aligned}$$

- $\omega(P)$ is piecewise linear, and convex in price P at each subinterval;

Properties

$$\omega(P) = \min_{\alpha}\{c(V, m(V,P)) - \alpha(V) : \alpha(s) \le c(s, m(s,P)), \forall s \in S, \alpha \in \mathbb{R}^{v}\};$$

- $\omega(P)$ is piecewise linear, and convex in price P at each subinterval;
- $\omega(P)$ can be bounded by zero when the number of using machines, m, is larger than $\frac{n}{2}$.

Properties

$$\omega(P) = \min_\alpha\{c(V, m(V,P)) - \alpha(V) : \alpha(s) \leq c(s, m(s,P)), \forall s \in S, \alpha \in \mathbb{R}^v\};$$

- $\omega(P)$ is piecewise linear, and convex in price P at each subinterval;
- $\omega(P)$ can be bounded by zero when the number of using machines, m, is larger than $\frac{n}{2}$.
- $P_i, 2 \leq i \leq v$ can be obtained by SPT rules.

Properties

$$\begin{aligned}\omega(P) &= \min_\alpha\{c(V, m(V, P)) - \alpha(V) : \\ &\alpha(s) \leq c(s, m(s, P)), \forall s \in S, \alpha \in \mathbb{R}^v\};\end{aligned}$$

- $\omega(P)$ is piecewise linear, and convex in price P at each subinterval;
- $\omega(P)$ can be bounded by zero when the number of using machines, m, is larger than $\frac{n}{2}$.
- $P_i, 2 \leq i \leq v$ can be obtained by SPT rules.
- $P_1 = P_2 + \cdots + P_n = \sum_{i=2}^{n} P_i$.

Properties

- When the number of using machines is 1 for the grand coalition, the range of slopes of the line segments in the interval is $\left(-1, -\frac{1}{n-1}\right]$, and the number of breakpoints is $O(v^2)$;

Properties

- When the number of using machines is 1 for the grand coalition, the range of slopes of the line segments in the interval is $\left(-1, -\frac{1}{n-1}\right]$, and the number of breakpoints is $O(v^2)$;
- Define that

$$\begin{aligned}\omega_1(P) = \quad & \min_\alpha \{c(V, m(V, P)) - \alpha(V) : \\ & \alpha(s) \le c(s) + P, \forall s \in S, \alpha \in \mathbb{R}^v\}\end{aligned}$$

Then the original problem $\omega(P)$ is equivalent to $\omega_1(P)$ which means that all sub-coalitions only use one machine.

Algorithms & Computations

IPC Algorithm

The Intersection Points Computation(IPC) Algorithm to Construct $\omega(P)$ Function.

Step 1. Initially, set $I^* = \{P_L, P_H\}$ and $\mathbb{I} = \{[P_L, P_H]\}$.

Step 2. When $\mathbb{I}$ is not empty: Sort values in I^* by $P_0 < P_1 < \cdots < P_q$, where $P_0 = P_L, P_q = P_H$ and $q = |I^*| - 1$.

Step 3. Obtain an LPB cost allocation α_{LP}^{GF} with $\alpha_{LP}^{GF}(N) = \pi_{\mathrm{LP}}(N)$.

CP Algorithm

The Intersection Points Computation(IPC) Algorithm to Construct $\omega(P)$ Function.

Step 1. Initially, set $I^* = \{P_L, P_H\}$ and $\mathbb{I} = \{[P_L, P_H]\}$.

Step 2. When $\mathbb{I}$ is not empty: Sort values in I^* by $P_0 < P_1 < \cdots < P_q$, where $P_0 = P_L, P_q = P_H$ and $q = |I^*| - 1$.

Step 3. Obtain an LPB cost allocation α_{LP}^{GF} with $\alpha_{LP}^{GF}(N) = \pi_{\mathrm{LP}}(N)$.

DP Algorithm

The Intersection Points Computation(IPC) Algorithm to Construct $\omega(P)$ Function.

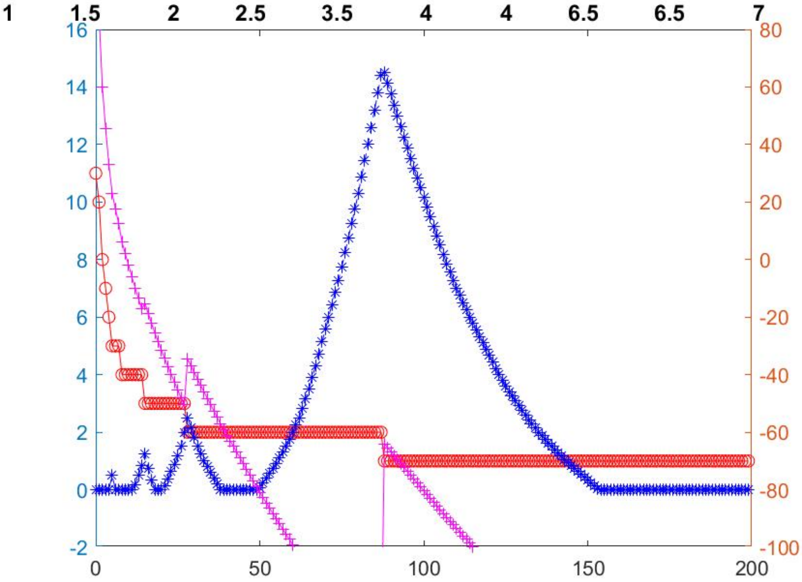
Step 1. Initially, set $I^* = \{P_L, P_H\}$ and $\mathbb{I} = \{[P_L, P_H]\}$.

Step 2. When $\mathbb{I}$ is not empty: Sort values in I^* by $P_0 < P_1 < \cdots < P_q$, where $P_0 = P_L, P_q = P_H$ and $q = |I^*| - 1$.

Step 3. Obtain an LPB cost allocation α_{LP}^{GF} with $\alpha_{LP}^{GF}(N) = \pi_{\mathrm{LP}}(N)$.

Computational Results

Image



1 1.5 2 2.5 3.5 4 4 6.5 6.5 7
16 14 12 10 8 6 4 2 0 -2
80 60 40 20 0 -20 -40 -60 -80 -100
0 50 100 150 200

Extension & Generalization

Machine Scheduling Game with Weighted Jobs

Pricing in General IM Games

Conclusions

⋆ Cooperative Game Theory:

– New Instrument for Stabilization via Setup cost Pricing.

⋆ Scheduling Problem:

– Constrained Inverse Optimization Problem.

⋆ Models, Solution Methods and Applications:

– Several equivalent LP formulations;

– Feasibility analyses & How to handle infeasibility;

– Implementations on WMG and UFL games.

The End

Thank you!