Summary

Dis·*count*

August 21, 2019

Introduction

At first, we will introduce some preliminary knowledge about cooperative game theory as follows. A pair (V, c) is usually used to represent a cooperative game with transferable utilities, among which $V = \{1, 2, \ldots, v\}$ denotes a set of v players and $c : 2^V \to R$ indicates the characteristic function of the game. A coalition is defined as a non-empty subset of players; while V is refered to as a coalition, $S = 2^V \setminus \{\emptyset\}$ denotes the set of all feasible coalitions. The characteristic function of the game, $c(s)$, represents the minimum coalitional cost the players in s have to pay in order to cooperate together. A cost allocation vector $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_v] \in R^v$ is required by the game to maintain cooperation in the grand coalition, and α_k is the cost assigned to players $k \in V$ to assure no individual or group of players has the incentive to deviate. For the convenience of abbreviation, we use $\alpha(s) = \sum_{k \in s} \alpha_k$ to denote the total cost assigned to the players in coalition s. One of the most important concepts in cooperative game theory is core, which is a cost allocation satisfying two kinds of contraints, one is the budget balance contraint $\alpha(V) = c(V)$ and the other is the coalitional stability constrains $\alpha(s) \leq c(s)$. In other words, core can be expressed as

$$Core(V, c) = \{\alpha : \alpha(V) = c(V), \alpha(s) \leq c(s) \quad s \in S \setminus \{V\}, \alpha \in R^v\}.$$

When the cost allocation exists, core is called non-empty. And if and only if the core is non-empty, the grand coalition of the associating cooperative game (V, c) will be stable or balanced.

However, cooperative games can be unbalanced in many cases owing to the joint restrictions of the above-mentioned two kinds of constraints. To stabilize the grand coalitions in unbalanced cooperative games, researchers have already developed several effective instrument, such as subsidization, penalization and simultaneous subsidization and penalization. The similarities of these instruments is that there exists an outside party who will take measures to stabilize the grand coalition. But the penalization will always arouse the discontent of players in coalitions, it's promising to find a new instrment to replace the penalization, we call it taxation. The significant idea of this instrument is that we can erase the role of the third party by collecting taxation as subsidy of the corresponding coalitions. That being said, the players of the games can stabilize themselves without the third party.

To make the project concrete, we will apply the cooperative theory on the machine schedule problem to show our ideas.

Then we will illustrate the taxation instrument with a simple example of an IPU (identical parallel machine scheduling of unweighted jobs) game as follows.

Let $N = 1, 2, \ldots, n$ be a set of n players. The number of machines is m and the setup cost is t_0. For convenience of expression, we set the processing times $t_i, i \in N$ satisfy $t_1 < t_2 < \cdots < t_n$.

1 Example

There is a IPU game which contains four players, whose processing times on the identical parallel machine are $t_1 = 2, t_2 = 3, t_3 = 4, t_4 = 5$ respectively. Each machine setup cost is $t_0 = 9.5$, and $c(s)$ is the minimum total completion time of jobs in coalition s plus the machine setup cost.

In this example, the grand coalition has a minimum cost of 38, and the optimum number of machine is 2. It's easy to check that this example is unbalanced and calculate the subsidy which equals $c(V) - \alpha(V) = 0.75$.

When the setup cost increases from 9.5 to 13, that is, the taxation increases from 0 to 3.5, the number of machine will decrease from 2 to 1.

2 Conclusion

Now we extend the number of players to a more complex case, that is we will set the number of players to n. In this situation, the interval size of setup cost can be calculated when t_i is known. And we can obtain that under what circumstances the machine number will change. We set the different intervals where the number of machines remain unchanged as I_i respectively. The extreme points of these intervals are recorded as S_i respectively. Specially, S_1 denotes the extreme point of the whole interval, which means the value of setup cost when subsidy equals zero.

We will demonstrate the three parts of the graph.

Because the costs of all the exponential coalitions can be easily got by the SPT rules, we have Lemma1

3 Lemma1

The value at the extreme points of the sub-intervals I_i can be calculated with processing times t_i by comparing the costs of the grand coalitions where all the players use two adjacent numbers of machines.

According to the above lemma1, we can obtain all the extreme points of the sub-intervals, i.e., the number of using machines decreases by one when the setup cost equals S_i.

4 Lemma2

The characteristic function must be supermodular, then we can get the positive setupcost when the machine number changes.

5 Corollary1

According to the foregoing description, we have the equation $S_1 = S_2 + \cdots + S_n = \sum_{i=2}^{n} S_i$.

With corollary1, we obtain all the breakpoints during the interval of the setup cost where the machine number changes and the subsidy is 0. Then we'll focus on the specific property of subsidy.

6 Corollary2

The least absolute value of the slope during all the intervals is $\frac{1}{n-1}$. The values of the slope during the sub-intervals are proper fraction. Meanwhile the sum of numerator and denominator is no more than n.

7 Corollary3

The subsidy is always zero when m is larger than $\frac{n}{2}$. In other words, when the numbers of machine is larger than half of players, the grand coalition don't need any subsidy from the externality.

Until here, we described the main property of the whole figure. And a diagram of the number of machines and subsidy on setup cost, with its essential features is showed below.

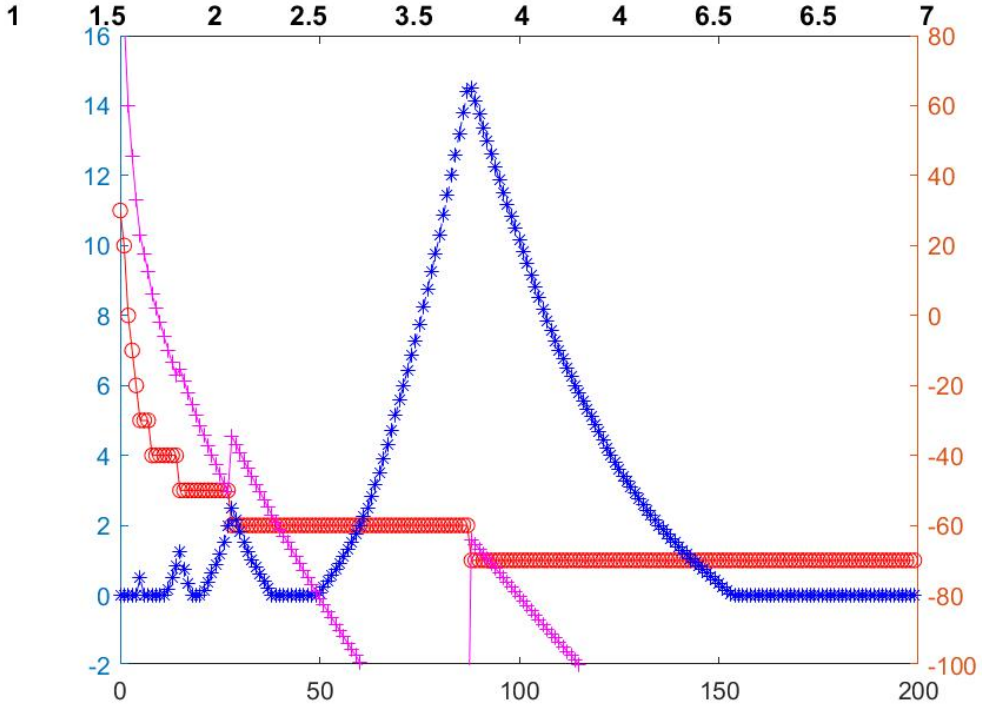

Figure 1: The number of machines and subsidy on setup cost.

The processing times of all players with the arrangement from small to large are listed on the top of this figure. The red and blue lines stand for the machine number and subsidy,respectively. The horizontal ordinate represents setup cost. The left ordinate represents the value of machine number while the right represents the value of subsidy.

8 Proof1

For convenience of expression, we set the setup cost as $S_1, S_2, \ldots, S_n$ at interval points while the number of machine changes. And S_i denotes the setup cost when the machine number changes from i to $i-1$, especially, S_1 denotes the least setup cost when machine number is 1 and the corresponding subsidy is 0. We have the equality

$$S_1 = S_2 + \cdots + S_n = \sum_{i=2}^{n} S_i.$$

Notice that

$$(n-1) \sum_{s \in S \setminus \{V\}} \rho_s \geq \sum_{k \in V} \sum_{s \in S \setminus \{V\}: k \in s} \rho_s = n.$$

The left side of the inequality means for every ρ_s can appear at most $(n-1)$ times, so we should know that if and only if for every $\rho_s > 0$ appears $n-1$ times the quality holds.That is to say, the coalitions which contains

$n-1$ players are all maximally unsatisfied coalitions. Then we have $\binom{n}{n-1}$ equalities.

$$\begin{cases} \alpha_1 + \alpha_2 + \cdots + \alpha_{n-1} & = x_1 \\ \alpha_1 + \alpha_3 + \cdots + \alpha_n & = x_2 \\ \quad \vdots & \vdots \\ \alpha_2 + \alpha_3 + \cdots + \alpha_n & = x_n. \end{cases}$$

Add these n equations together, and we can get

$$(n-1)(\alpha_1 + \alpha_2 + \cdots + \alpha_n) = \sum_{i=1}^{n} x_i$$

As we know, $x_1, x_2, \ldots, x_n$ can be expressed as follows:

$$\begin{cases} x_1 = S_0 + (n-1)t_1 + (n-2)t_2 + & \cdots + t_{n-1} \\ x_2 = S_0 + (n-1)t_1 + (n-2)t_3 + & \cdots + t_{n-1} \\ \quad \vdots & \vdots \\ x_n = S_0 + (n-1)t_2 + (n-2)t_3 + & \cdots + t_n \end{cases}$$

According to SPT rule, we can obtain the equality $c(V) = \alpha_1 + \alpha_2 + \cdots + \alpha_n = S_0 + nt_1 + (n-1)t_2 + \cdots + t_n$. By replacing $x_1, x_2, \ldots, x_n$ together with the expression of $c(V)$, we can get a equality only with $S_0, x_1, x_2, \ldots, x_n$.

Finally, we can obtain $S_0 = \sum_{k=1}^{n}(n-k)t_k$.

9 Proof2

10 Suppose1

This one is about how to predict the slope of machine scheduling.

As we konw,

11 Suppose2