Tilburg University

Processing games with shared interest

Quant, M.; Meertens, M.; Reijnierse, J.H.

Link to publication in Tilburg University Research Portal

Processing games with shared interest

Marieke Quant · Marc Meertens · Hans Reijnierse

Abstract This paper introduces processing problems with shared interest as an extension of processing situations with restricted capacities (Meertens, M., et al., Processing games with restricted capacities, 2004). Next to an individual capacity to handle jobs, each player now may have interest in the completion of more than one job, and the degrees of interest may vary among players. By cooperating the players can bundle their capacities and follow an optimal processing scheme to minimize total joint costs. The resulting cost allocation problem is analyzed by considering an associated cooperative cost game. An explicit core allocation of this game is provided.

Keywords Processing games · Scheduling · Core allocation

1 Introduction

Consider the situation in which a number of jobs need to be completed by a number of players. Each player is endowed with an *individual capacity* for handling jobs. Each job requires a certain amount of effort (i.e., it has a *processing demand*) and for each time unit a job is uncompleted it generates a fixed cost (called the *cost-coefficient* of the job).

In Meertens et al. (2004) these so-called processing situations are studied in a cooperative game theory framework. They consider a model in which a player is assigned to *exactly one* job (i.e., there is a one-one correspondence between the jobs and the players). If a coalition of players cooperate, costs savings can be made by helping each other by means of using a player's capacity to speed up the completion of a job of another coalition member. So, a coalition has the sum of the capacities of its members to its disposal for completing all jobs of the coalition in such a way that the total costs are minimized. To minimize the total costs

M. Quant (✉) · M. Meertens · H. Reijnierse
CentER and Department of Econometrics & OR, Tilburg University, P.O. Box 90153, 5000, LE Tilburg, The Netherlands
e-mail: quant@uvt.nl

in a processing problem, jobs have to be completed one by one according to the order of decreasing *urgencies* (cf. Smith 1956).

At first sight, this observation might lead to the idea that the cooperative games derived from processing situations are equivalent to *sequencing games* with one machine and aggregated weighted completion times (cf. Curiel et al. 1989). However, this is *not* the case. The diversion between these two classes of games is due to two main differences between a processing situation and a sequencing situation with one machine and aggregated weighted completion times. The first difference is that players have individual, and in general, different capacities for handling jobs (not necessarily their own), while in a sequencing situation with one machine, there is a *constant capacity* (the capacity of the machine) for handling jobs. The second difference is that in a sequencing situation there is an *initial scheduling* of the jobs in front of the machine. In a processing situation, however, there are *no* initial restrictions nor rights on the order in which players may want to process their jobs.

The main result in Meertens et al. (2004) shows total balancedness of the cost-games derived from processing situations by providing an explicit core element. In this paper, we extend this result to a wider class of processing situations. These extensions are similar to the ones that can be found in recent literature on sequencing situations. There, the assumption that each player is obliged to one job has been dropped and the idea that a job can be of interest for different players has been introduced (see Estévez-Fernández et al. 2004 or Calleja et al. 2004).

Here, we study processing situations with *shared interest*. In a processing situation with shared interest each player has again a individual capacity for completing jobs which have certain processing demands. However, a player may have *interest* for *more than one* job (or even for all jobs) to be completed. Moreover the cost-coefficients do *not* only depend on the jobs, but also on the players. So, a job may be of interest for several players, but the costs for the time this job is uncompleted may be *different* for each player involved. The cost-games derived from processing situations with shared interest are totally balanced. In fact, we provide an explicit core element for these type of cost-games.

The paper is organized as follows. In Sect. 2 we briefly repeat the results of Meertens et al. (2004). Section 3 introduces the new model and provides an explicit core element. Section 4 contains the conclusions.

2 Processing games

A *processing problem* $\mathcal{P}$ with restricted capacity consists of a tuple

$$\langle J, p = (p_j)_{j \in J}, \alpha = (\alpha_j)_{j \in J}, \beta \rangle.$$

Here, J is a finite set of jobs that need to be completed. The vector $p \in \mathbb{R}_{++}^J$ contains the *processing demands* or efforts of the jobs, furthermore $\alpha \in \mathbb{R}_+^J$ is the vector of *cost coefficients* and β is a strictly positive real number denoting the maximum available effort per time unit, or shortly *capacity*. The costs for job $j \in J$ to be uncompleted for a period of time t equals $\alpha_j \cdot t$. The objective is to find a feasible schedule such that the total costs are minimized. This minimum is denoted by $c(\mathcal{P})$. To attain these minimal costs, the jobs should be completed one after an other. The order in which this is done depends on the urgency of the jobs. This leads to the following proposition.

Proposition 2.1 (cf. Smith 1956) *Let $\mathcal{P}$ be a processing problem such that $J = \{1, \ldots, |J|\}$ and the jobs are numbered such that $\frac{\alpha_1}{p_1} \geq \cdots \geq \frac{\alpha_{|J|}}{p_{|J|}}$. Then it is optimal to process the jobs*

in increasing order and

$$c(\mathcal{P}) = \frac{1}{\beta} \sum_{i=1}^{|J|} \alpha_i \cdot [p_1 + \cdots + p_i].$$

In a *processing situation* $\langle N, J, p = (p_j)_{j \in J}, \alpha = (\alpha_j)_{j \in J}, (\beta_i)_{i \in N} \rangle$ there is a finite set of *players* N, in which each player $i \in N$ is endowed with a strictly positive *capacity* of β_i in order to perform jobs. Each *job* $j \in J$ has a *processing demand* $p_j > 0$ and cost coefficient $\alpha_j \geq 0$. As long as job j is uncompleted, it generates a cost of size α_j per time unit. It is assumed that each job is assigned to a different player. Therefore, we denote the processing demand and the cost coefficient of the job of player i by p_i and α_i respectively.

Let $S \subseteq N$ be a coalition of players who decide to cooperate. This coalition has the individual capacities of its members to its disposal, i.e. coalition S can maximally generate an amount of effort of size $\beta(S) := \sum_{i \in S} \beta_i$ per time unit. The aim of coalition S is to complete all jobs of its members, such that aggregate costs are minimized. This situation gives rise to the processing problem

$$\mathcal{P}(S) := \langle J(S), (p_i)_{i \in S}, (\alpha_i)_{i \in S}, \beta(S) \rangle,$$

in which $J(S)$ denotes the set of jobs of players in S. The *processing game* $\langle N, c^{\mathcal{P}} \rangle$ in which $c^{\mathcal{P}} : 2^N \to \mathbb{R}_+$ is the map defined by

$$c^{\mathcal{P}}(S) := c(\mathcal{P}(S)) \quad \text{for all } S \subseteq N.$$

Theorem 2.1 (Meertens et al. 2004) *Processing games are totally balanced.*

Let $\langle N, J, p = (p_j)_{j \in J}, \alpha = (\alpha_j)_{j \in J}, (\beta_i)_{i \in N} \rangle$ be a processing situation. In fact the authors show that the vector $y \in \mathbb{R}^N$ with

$$y_i = \frac{\alpha_i}{\beta(N)} \sum_{k=1}^{i} p_k + \tau_i - \frac{\beta_i}{\beta(N)} \sum_{k=1}^{n} \tau_k, \tag{1}$$

for all $i \in N$, is a core allocation of the corresponding processing game, provided that $N := \{1, \ldots, n\}$ and $\frac{\alpha_1}{p_1} \geq \cdots \geq \frac{\alpha_n}{p_n}$. Here τ_i denotes the tax paid by player i, and this tax is defined by:

$$\tau_i := \frac{p_i}{\beta(N)} \cdot \left[\frac{1}{2} \cdot \alpha_i + \sum_{k=i+1}^{n} \alpha_k \right].$$

So, the amount player $i \in N$ has to pay in the core allocation y consists of three parts. The first part $\frac{\alpha_i}{\beta(N)} \sum_{k=1}^{i} p_k$ gives the *actual costs* of player i. These are the product of the cost coefficient of player i and the completion time of his job. The second part τ_i is the tax paid by player i, which is paid because all players work on the job of player i and all jobs with lower urgencies have to wait for their completion. The sum of taxes is redivided among the players proportionally to their capacities. This results in the third part $-\frac{\beta_i}{\beta(N)} \sum_{k=1}^{n} \tau_k$, called the *subsidy* to player i. It is done in order to reward players with large capacity and compensate players of which the job is at the end of the line. A more detailed explanation of this core element can be found (Meertens et al. 2004).

3 Processing games with shared interest

In this section we analyze an extension of processing situations and the corresponding processing games. Instead of each player having *one* job, we allow them to have interest in several jobs. We prove that the corresponding games are totally balanced.

A *processing situation with shared interest* P can be described by a tuple $\langle N, J, p = (p)_{j \in J}, A, \beta = (\beta_i)_{i \in N} \rangle$. Here, N is a finite set of players and J a finite set of jobs. The vector $p \in \mathbb{R}^J_{++}$ contains the processing demands of the jobs. β contains the capacities with which the players are endowed. The matrix $A \in \mathbb{R}^N_+ \times \mathbb{R}^J_+$ contains the cost coefficients of all players for all possible jobs. The number A_{ij} denotes the cost coefficient of player $i \in N$ with respect to job $j \in J$. If $A_{ij} = 0$ then player i has no interest in job j. Contrary to the original setting it is now possible for a player to have interest in several jobs and a job can be of interest for more than one player. The original problem can be modeled as a processing situation with shared interest by choosing the matrix A to be a diagonal matrix. Let $i \in N$. The set of jobs in which player i is interested is denoted by $J_i = \{j \in J \mid A_{ij} > 0\}$.

If a coalition $S \subseteq N$ decides to cooperate, its members have a total capacity of $\beta(S) := \sum_{i \in S} \beta_i$ available in order to construct a schedule which completes all their jobs, such that total weighted costs are minimized. This situation gives rise to the processing problem

$$\mathcal{P}(S) := \langle J(S), (p_j)_{j \in J(S)}, (A_j(S))_{j \in J(S)}, \beta(S) \rangle,$$

in which $J(S)$ denotes $\cup_{i \in S} J_i$, the set of all jobs which are of interest for players of S. $\sum_{i \in S} A_{ij}$ is the total cost coefficient of coalition S for job j and is denoted by $A_j(S)$. Analogous to the problem in which each player has only one job, one can associate a *processing game with shared interest* $\langle N, c^P \rangle$ with $c^P : 2^N \to \mathbb{R}_+$ as follows:

$$c^P(S) := c(\mathcal{P}(S)) \quad \text{for all } S \subseteq N.$$

Next we provide an example of a processing situation with shared interest, derived from a maintenance problem.

Example 3.1 Suppose a tree network is given, that connects all players to the root. The players are situated in the nodes of the tree. Over time the network has deteriorated and the players suffer from periodic costs (e.g., leakage, delay because of bad roads). One day they decide to collectively perform a renovation. Each edge j in the network needs a specific amount of effort p_j to be renovated, e.g., one worker needs p_j days to fix road j. Each player i can contribute β_i workers. Each player i suffers an amount of costs equal to A_{ij} for each day that edge j has not been repaired yet. Note that A_{ij} is positive only if edge j is along the path of node i to the root. The players have to decide in which order the edges will be repaired. Furthermore, players with relatively high costs on edges that are repaired in a late stage, or who contribute more workers have to be compensated. The situation described can be modeled by a processing game with shared interest. Let us illustrate this further by means of a numerical example.

Suppose we have two players and three edges, one of them used by both. The efforts for renovating the edges equal $p := (30, 15, 20)$, *i.e.* one worker needs p_j days to repair edge j (see Fig. 1).

Player 1 contributes one worker, while player 2 can contribute two workers. The costs that each player suffers for each day an edge is not yet fixed, are reflected in the following matrix

$$A = \begin{pmatrix} 3 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix}.$$

Fig. 1 Tree with two players. The *black nodes* (including the root) are vacant
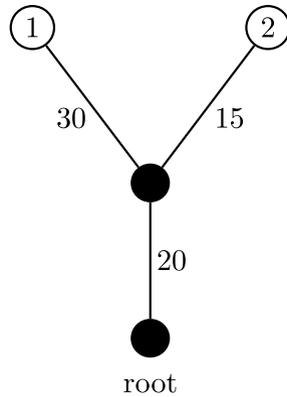
Row i denotes the costs of player i with respect to the edges. If player 1 works by himself, then he first renovates the edge with requires an effort of 30 and finally the common edge. Since, he is all by himself (i.e., $\beta_1 = 1$) this yields a total costs of $c^P(\{1\}) = 3 \cdot \frac{30}{1} + 1 \cdot \frac{20+30}{1} = 140$. Similarly, the costs for player 2 (note that $\beta_2 := 2$) equals $c^P(\{2\}) = 2 \cdot \frac{20}{2} + 1 \cdot \frac{15+20}{2} = 37\frac{1}{2}$. If both players decide to cooperate they can contribute three workers for the renovations. Although the common edge does not require the most effort, it should be repaired first (it is of interest for both players), then the private edge of player 1 and finally the private edge of player 2. The total minimal joint costs equal

$$c^P(N) = 3 \cdot \frac{20}{3} + 3 \cdot \frac{20+30}{3} + 1 \cdot \frac{20+30+15}{3} = 91\frac{2}{3}.$$

The main result of this paper is stated in the following theorem.

Theorem 3.1 *A processing game with shared interest is balanced.*

In the proof of Theorem 3.1 an explicit core allocation is provided. We first introduce this core allocation x and give an intuitive explanation.

Let $P = \langle N, J, p, A, \beta \rangle$ be a processing situation with shared interest, such that $J = \{1, \ldots, |J|\}$ and $\frac{A_1(N)}{p_1} \geq \cdots \geq \frac{A_{|J|}(N)}{p_{|J|}}$ (this means that if the grand coalition cooperates, the jobs are performed in the order $(1, \ldots, |J|)$). Then

$$x_i := \left(\sum_{j \in J} \frac{A_{ij}}{\beta(N)} \sum_{k=1}^{j} p_k \right) + \left(\sum_{j \in J} \frac{A_{ij}}{A_j(N)} \tau_j \right) - \frac{\beta_i}{\beta(N)} \sum_{j \in J} \tau_j \qquad (2)$$

for all $i \in N$. The value τ_j, is for every $j \in J$ defined by

$$\tau_j = \frac{p_j}{\beta(N)} \left[\frac{1}{2} A_j(N) + \sum_{k=j+1}^{|J|} A_k(N) \right].$$

The division of the allocation into three parts is still accurate. The first part concerns the actual costs of the jobs. Each player i has to pay his personal actual costs of each job j. The second part gives the taxes of the jobs. These are imposed in proportion with the costs

coefficients. The third part concerns the subsidies, which are still allocated proportionally with respect to the capacities of the players.

The core element x is independent on the optimal order chosen. Namely, (2) can be rewritten as

$$x_i := \sum_{j \in J} \frac{A_{ij}}{A_j(N)} \left(\frac{A_j(N)}{\beta(N)} \sum_{k=1}^{j} p_k + \tau_j \right) - \frac{\beta_i}{\beta(N)} \sum_{j \in J} \tau_j. \tag{3}$$

The independence now follows immediately from Meertens et al. (2004), since they prove that if another optimal order is used, the total amount of taxes paid does not change, and neither does the term $\frac{A_j(N)}{\beta(N)} \sum_{k=1}^{j} p_k + \tau_j$ for each $j \in J$.

We now turn to the proof of Theorem 3.1. We first prove a lemma on balancedness for *processing situations with multiple jobs* using a processing game in the original setting. In a processing situation with multiple jobs players can have interest in more than one job, but each job is still of interest for only one player. In fact each player "owns" a set of jobs. Subsequently, the proof of Theorem 3.1 is built along the following lines: first it is shown that it is sufficient to prove that the allocation x as described in formula (2) is a core allocation in case $|N| = 2$. Then we prove that x is a core element for all processing situations in which $|N| = 2$. The proof is given by an induction argument on the number of jobs for which both players have a positive cost coefficient. Note that if this number equals zero, we have a multiple jobs processing situation of which a core element is already provided. In the induction step, a job with joint interest is split into two jobs, both with single interest. With the help of this new processing situation on which the induction hypothesis can be applied (and hence a core element is known), the result is proved.

Lemma 3.1 *Every multiple jobs processing game is totally balanced.*

Proof Let (N, c^P) be a multiple jobs processing game with associated situation $P = \langle N, J, p, A, \beta \rangle$, i.e. $J_i \cap J_k = \emptyset$ for all $i, k \in N$ with $i \neq k$. Define the processing situation $\bar{\mathcal{P}} = \langle J(N), J, p, \bar{\alpha}, \bar{\beta} \rangle$ as follows. For each job in J a player is created, so the player set of $\bar{\mathcal{P}}$ is $J(N)$ and $J(N) = J$. A 'player' j, which is actually a job, keeps its processing demand p_j and its cost coefficient becomes $\bar{\alpha}_j = A_{ij} > 0$, in which i is the unique player such that $A_{ij} > 0$. The capacity β_i of a regular player i in N is split equally over all new players originating from his job set J_i, resulting in:

$$\bar{\beta}_j := \frac{\beta_i}{|J_i|} \quad \text{for all } i \in N, \ j \in J_i.$$

It is left to the reader to verify that for each coalition $S \subseteq N$ the following is true,

$$c^P(S) = c^{\bar{\mathcal{P}}}(J(S)). \tag{4}$$

According to Theorem 2.1, the game $\langle J(N), c^{\bar{\mathcal{P}}} \rangle$ is totally balanced and has a core allocation y in $\mathbb{R}^{J(N)}$, see (1). Define x in $\mathbb{R}^N$ by $x_i := \sum_{j \in J_i} y_j$ for all i in N. Since equation (4) is valid and $y \in \mathcal{C}(c^{\bar{\mathcal{P}}})$, it follows that $x \in \mathcal{C}(c^P)$. $\qquad\square$

Proof of Theorem 3.1 Let $P = \langle N, J, p, A, \beta \rangle$ be a processing situation with shared interest. Without loss of generality we assume that $J = \{1, \ldots, |J|\}$ and $\frac{A_1(N)}{p_1} \geq \cdots \geq \frac{A_{|J|}(N)}{p_{|J|}}$. Let x be the allocation as described in (2).

Step 1: in this step we show that is sufficient to prove that x is a core allocation if $|N| = 2$.

Suppose that P is such that $x \notin C(c^P)$. Then there exists a coalition $S \subset N$ such that $x(S) = \sum_{i \in S} x_i > c^P(S)$. We construct a new processing situation $\bar{P}$, with player set $\bar{N} = \{1, 2\}$ and show that for this new situation the corresponding allocation $\bar{x}$ is not a core element.

Define a two person processing situation $\bar{P}$ with shared interest: $\bar{P} = \langle \bar{N} = \{1, 2\}, J, p, \bar{A}, \bar{\beta} \rangle$. The job set and the vector of processing demands remain unchanged. Player 1 can be seen as a representative of coalition S and player 2 as a representative of coalition $N \backslash S$. The matrix of cost coefficients $\bar{A}$ is defined as follows: the cost coefficient $\bar{A}_{1j}$ of player 1 and job j equals the total cost coefficient for coalition S: $\bar{A}_{1j} = A_j(S)$. Similarly, the cost coefficient $\bar{A}_{2j}$ of player 2 and job j equals the total cost coefficient for coalition $N \backslash S$: $\bar{A}_{2j} = A_j(N \backslash S)$. The vector of capacities $\bar{\beta}$ is given by $\bar{\beta} = (\beta(S), \beta(N \backslash S))$. Note that the coalitions N and $\bar{N}$ perform exactly the same jobs and for each job $j \in J$, $A_j(N) = \bar{A}_j(\bar{N})$. Hence, the optimal orders of the problems coincide. The same is true for the coalitions S and $\{1\}$, and $N \backslash S$ and $\{2\}$. Hence

$$c^P(N) = c^{\bar{P}}(\bar{N}), \qquad c^P(S) = c^{\bar{P}}(\{1\}), \qquad c^P(N \backslash S) = c^{\bar{P}}(\{2\}).$$

Let $\bar{x}$ be the allocation corresponding to $\bar{P}$ (as described in (2)). Since $\bar{A}_{1j} = \sum_{i \in S} A_{ij}$ and $\bar{A}_{2j} = \sum_{i \in N \backslash S} A_{ij}$, it follows from equation (3) that

$$x(S) = \bar{x}_1,$$

$$x(N \backslash S) = \bar{x}_2$$

This indicates that $\bar{x}$ is not a core allocation of the game $\langle \bar{N}, c^{\bar{P}} \rangle$. Hence, if there exists a situation in which x is not a core allocation, then there exists also such a situation with only two players.

Step 2: in this step we prove that for all processing situations with two players, the allocation x as described in (2) is a core element.

Assume that $|N| = 2$. Let $\ell(P)$ be the number of jobs for which both players have a strictly positive cost coefficient (i.e. $\ell(P) := \{j \in J \mid A_{1j} > 0 \text{ and } A_{2j} > 0\}$). We prove our statement by induction on $\ell(P)$.

If $\ell(P) = 0$ we deal with a processing problem with multiple jobs. In this case, x coincides with the allocation that is shown to be in the core of the corresponding processing game with multiple jobs in the proof of Lemma 3.1.

Let $k \in \mathbb{N}$. Assume that for each two person processing game with shared interest for which $\ell(P)$ does not exceed k, x is a core element. Let $P = \langle N = \{1, 2\}, J, p, A, \beta \rangle$ be a processing situation such that $\ell(P)$ equals $k + 1$. We prove that $x \in C(c^P)$, which completes the induction argument.

Let j be a job with shared interest. Without loss of generality we assume that the processing demand of job j equals the total cost coefficient: $p_j = A_j(N)$ (this is just a matter of scaling). So the urgency of job j equals 1. Define another processing situation $\bar{P} = \langle N, \bar{J}, \bar{p}, \bar{A}, \beta \rangle$ as follows. $\bar{P}$ arises from P by splitting j into two jobs j_a and j_b with single interest, both having urgency 1, like j. This yields

$$\bar{J} = \{1, \ldots, j-1, j_a, j_b, j+1, \ldots, |J|\},$$

$$\bar{p} = \{p_1, \ldots, p_{j-1}, A_{1j}, A_{2j}, p_{j+1}, \ldots, p_{|J|}\},$$

$$\bar{A} = \begin{pmatrix} A_{11} & \ldots & A_{1(j-1)} & A_{1j} & 0 & A_{1(j+1)} & \ldots & A_{1|J|} \\ A_{21} & \ldots & A_{2(j-1)} & 0 & A_{2j} & A_{2(j+1)} & \ldots & A_{2|J|} \end{pmatrix}.$$

Let $\bar{x}$ be the allocation defined by (2) that corresponds to $\bar{P}$. By the induction hypothesis it holds that $\bar{x} \in C(c^{\bar{P}})$. Hence $c^{\bar{P}}(\{i\}) - \bar{x}_i \geq 0$ for $i \in \{1, 2\}$. Note that it is sufficient to prove that $c^P(\{i\}) - x_i \geq c^{\bar{P}}(\{i\}) - \bar{x}_i$, since this guarantees that x is a core element. It is equivalent to show that

$$c^P(\{i\}) - c^{\bar{P}}(\{i\}) \geq x_i - \bar{x}_i. \tag{5}$$

In order to prove inequality (5), we prove two statements:

$$c^P(\{i\}) - c^{\bar{P}}(\{i\}) \geq \frac{A_{1j}A_{2j}}{\beta_i}, \tag{6}$$

$$x_i - \bar{x}_i = \frac{A_{1j}A_{2j}}{2\beta(N)} \tag{7}$$

for $i \in \{1, 2\}$. Inequality (5) immediately follows from inequality (6), equality (7) and the fact that $\beta_i < \beta(N)$.

Proof of inequality (6): let $i = 1$. Note that the job set of player 1 in the new situation can be written as $\bar{J}_1 = (J_1 \backslash \{j\}) \cup \{j_a\}$. Let σ be the optimal order of player 1 for the processing problem $\mathcal{P}(\{1\})$ and let $\bar{\sigma}$ be the same order as σ, but job j is replaced by j_a. This means that player 1 completes the jobs in the old order σ, but instead of job j, job j_a is performed. Order $\bar{\sigma}$ is a possible order to solve the processing problem $\bar{\mathcal{P}}(\{1\})$. We compare the costs of σ with the costs of $\bar{\sigma}$. Since the processing demands of jobs j and j_a are not equal (contrary to their cost coefficients!), this yields a cost reduction of at least

$$\frac{1}{\beta_1}(p_j - p_{j_a}) \cdot A_{1j} = \frac{1}{\beta_1}(A_j(N) - A_{1j})A_{1j} = \frac{A_{1j}A_{2j}}{\beta_1}.$$

Note that the ready times of jobs beyond j become smaller in $\bar{\sigma}$, which yields an extra cost reduction. Hence the value found above is a lower bound for the cost reduction. It is possible that for player 1 another order becomes optimal when facing $\bar{\mathcal{P}}(\{1\})$, so:

$$c^{\bar{P}}(\{1\}) \leq c(\bar{\sigma}) \leq c^P(\{1\}) - \frac{A_{1j}A_{2j}}{\beta_1},$$

where $c(\bar{\sigma})$ denotes the costs if player $\{1\}$ performs his job in the order $\bar{\sigma}$. The same argument holds for player 2, which proves inequality (6).

Proof of inequality (7): jobs j_a and j_b have equal urgency for coalition N. Since x does not change if jobs with the same urgency are switched, we assume that the order corresponding to $\bar{x}$ equals $(1, \ldots, j-1, j_a, j_b, j+1, \ldots, |J|)$. Let $\bar{\tau}$ be the vector of taxes corresponding with $\bar{x}$. Before calculating the difference between the two allocations, we first show that the total amount of taxes paid in the new situation, equals the amount of taxes paid in the old situation: $\sum_{k \in \bar{J}} \bar{\tau}_k = \sum_{k \in J} \tau_k$.

Denote $\sum_{k > j} A_k(N)$ by s. The following equations are valid,

$$\bar{\tau}_k = \tau_k \quad \text{for all } k \in \bar{J} \backslash \{j_a, j_b\},$$

$$\bar{\tau}_{j_a} = \frac{A_{1j}}{\beta(N)}\left(\frac{1}{2}\bar{A}_{j_a}(N) + \bar{A}_{j_b}(N) + s\right),$$

$$\bar{\tau}_{j_b} = \frac{A_{2j}}{\beta(N)}\left(\frac{1}{2}\bar{A}_{j_b}(N) + s\right).$$

Because $\bar{A}_{j_a}(N) = A_{1j}$, $\bar{A}_{j_b}(N) = A_{2j}$ and $p_j = A_{1j} + A_{2j}$, we have

$$\bar{\tau}_{j_a} + \bar{\tau}_{j_b} = \frac{1}{\beta(N)}\left(\frac{1}{2}A_{1j}^2 + A_{1j}A_{2j} + A_{1j}s + \frac{1}{2}A_{2j}^2 + A_{2j}s\right)$$

$$= \frac{1}{\beta(N)}\left(\frac{1}{2}(A_{1j} + A_{2j})^2 + (A_{1j} + A_{2j})s\right)$$

$$= \frac{p_j}{\beta(N)}\left(\frac{1}{2}p_j + s\right) = \tau_j.$$

This implies that the total amount of taxes paid, does not change.

We can now calculate the difference between the two allocations x_1 and $\bar{x}_1$, and x_2 and $\bar{x}_2$. Recall that:

$$x_1 = \sum_{\ell \in J}\frac{A_{1\ell}}{\beta(N)}\sum_{k=1}^{\ell}p_k + \sum_{k \in J}\frac{A_{1k}}{A_k(N)}\tau_k - \frac{\beta_1}{\beta(N)}\sum_{k \in J}\tau_k,$$

$$\bar{x}_1 = \sum_{\ell \in \bar{J}}\frac{\bar{A}_{1\ell}}{\beta(N)}\sum_{k=1}^{\ell}\bar{p}_k + \sum_{k \in \bar{J}}\frac{\bar{A}_{1k}}{\bar{A}_k(N)}\bar{\tau}_k - \frac{\beta_1}{\beta(N)}\sum_{k \in \bar{J}}\bar{\tau}_k.$$

Since $\bar{p}_{j_a} + \bar{p}_{j_b} = p_j$, the completion times of all other jobs remain unchanged: $\bar{p}_k = p_k$ for all $k \in \bar{J}\backslash\{j_a, j_b\}$. Furthermore, recall that $\bar{A}_{1j_a} = A_{1j}$, $\bar{A}_{1j_b} = 0$ and $\bar{A}_{2j_a} = 0$, $\bar{A}_{2j_b} = A_{2j}$ and $p_j = A_{1j} + A_{2j}$, $\bar{p}_{j_a} = A_{1j}$ and $\bar{p}_{j_b} = A_{2j}$ and $\tau_j = \frac{A_j(N)}{\beta(N)}(\frac{1}{2}A_j(N) + s)$. Hence,

$$x_1 - \bar{x}_1 = \frac{A_{1j}}{\beta(N)}\sum_{k=1}^{j}p_k + \frac{A_{1j}}{A_j(N)}\tau_j - \frac{\bar{A}_{1j_a}}{\beta(N)}\sum_{k=1}^{j_a}\bar{p}_k - \frac{\bar{A}_{1j_a}}{\bar{A}_{j_a}(N)}\bar{\tau}_{j_a}$$

$$= \frac{A_{1j}}{\beta(N)}p_j - \frac{A_{1j}}{\beta(N)}\bar{p}_{j_a} + \frac{A_{1j}}{A_j(N)}\tau_j - \bar{\tau}_{j_a}$$

$$= \frac{A_{1j}A_{2j}}{\beta(N)} + \frac{A_{1j}}{\beta(N)}\left(\frac{1}{2}A_j(N) + s\right) - \frac{\bar{A}_{1j_a}}{\beta(N)}\left(\frac{1}{2}\bar{A}_{j_a}(N) + \bar{A}_{j_b}(N) + s\right)$$

$$= \frac{A_{1j}A_{2j}}{\beta(N)} + \frac{A_{1j}}{\beta(N)}\cdot\frac{1}{2}(A_{1j} + A_{2j}) - \frac{A_{1j}}{\beta(N)}\left(\frac{1}{2}A_{1j} + A_{2j}\right)$$

$$= \frac{A_{1j}A_{2j}}{2\beta(N)}.$$

Similarly,

$$x_2 = \sum_{\ell \in J}\frac{A_{2\ell}}{\beta(N)}\sum_{k=1}^{\ell}p_k + \sum_{k \in J}\frac{A_{2k}}{A_k(N)}\tau_k - \frac{\beta_2}{\beta(N)}\sum_{k \in J}\tau_k,$$

$$\bar{x}_2 = \sum_{\ell \in \bar{J}}\frac{\bar{A}_{2\ell}}{\beta(N)}\sum_{k=1}^{\ell}\bar{p}_k + \sum_{k \in \bar{J}}\frac{\bar{A}_{1k}}{\bar{A}_k(N)}\bar{\tau}_k - \frac{\beta_2}{\beta(N)}\sum_{k \in \bar{J}}\bar{\tau}_k.$$

And

$$
\begin{aligned}
x_2 - \bar{x}_2 &= \frac{A_{2j}}{\beta(N)} \sum_{k=1}^{j} p_k + \frac{A_{2j}}{A_j(N)} \tau_j - \frac{\bar{A}_{2j_b}}{\beta(N)} \sum_{k=1}^{j_b} \bar{p}_k - \frac{\bar{A}_{2j_b}}{\bar{A}_{j_b}(N)} \bar{\tau}_{j_b} \\
&= \frac{A_{2j}}{\beta(N)} p_j + \frac{A_{2j}}{A_j(N)} \tau_j - \frac{\bar{A}_{2j_b}}{\beta(N)} (\bar{p}_{j_a} + \bar{p}_{j_b}) - \bar{\tau}_{j_b} \\
&= \frac{A_{2j}}{A_j(N)} \tau_j - \bar{\tau}_{j_b} \\
&= \frac{A_{2j}}{A_j(N)} \left(\frac{A_j(N)}{\beta(N)} \left(\frac{1}{2} A_j(N) + s \right) \right) - \frac{A_{2j}}{\beta(N)} \left(\frac{1}{2} \bar{A}_{j_b}(N) + s \right) \\
&= \frac{A_{2j}}{\beta(N)} \left(\frac{1}{2} (A_{1j} + A_{2j}) + s \right) - \frac{A_{2j}}{\beta(N)} \left(\frac{1}{2} A_{2j} + s \right) \\
&= \frac{A_{1j} A_{2j}}{2\beta(N)}.
\end{aligned}
$$

This proves (7) and the theorem.   □

4 Conclusions

In this paper we introduce an extension of processing situations with restricted capacities, the so-called processing situations with shared interest. In this extension players may have interest for several jobs to be completed. We prove that the associated cost-games derived from these situations are totally balanced. This statement is proved by explicitly providing an allocation of the total costs which is a core element of the associated cost-game. Moreover, it is shown that this allocation is independent on the optimal order chosen to process the jobs.

References

Calleja, P., Estévez-Fernández, A., Borm, P., & Hamers, H. (2004). *Job scheduling, cooperation and control.* CentER DP 2004-65, Tilburg University, Tilburg, The Netherlands.

Curiel, I., Pederzoli, G., & Tijs, S. (1989). Sequencing games. *European Journal of Operational Research*, *40*, 344–351.

Estévez-Fernández, A., Borm, P., Calleja, P., & Hamers, H. (2004). *Sequencing games with repeated players.* CentER DP 2004-128, Tilburg University, Tilburg, The Netherlands.

Meertens, M., Borm, P., Quant, M., & Reijnierse, H. (2004). *Processing games with restricted capacities.* CentER DP 2004-83, Tilburg University, Tilburg, The Netherlands.

Smith, W. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, *3*, 59–66.