# Cost Sharing in a Job Scheduling Problem*

Debasis Mishra  Bharath Rangarajan
Center for Operations Research and Econometrics (CORE)
Université Catholique de Louvain
Email: {mishra,rangarajan}@core.ucl.ac.be

First version: February 2005; This version: April 2005

**Abstract**

A set of jobs need to be served by a server which can serve only one job at a time. Jobs have processing times and incur waiting costs (linear in their waiting time). The jobs share their costs through compensation using monetary transfers. In the first part, we provide an axiomatic characterization of the Shapley value rule by introducing some fairness axioms that are new in the literature. In the second part, we use linear programming duality to provide an alternate characterization of the Shapley value rule. Here, we use the idea of decomposition of transfers and the notion of *pairwise no-envy allocation*. Of the family of allocation rules that satisfy pairwise no-envy, the Shapley value rule is the one with the minimum sum of absolute values of transfers. We discuss no-envy rules and show that no-envy is not possible in general. If processing times of all jobs are equal, then it is possible to design no-envy rules, and we characterize all no-envy rules for this case.

**Keywords:** Queueing problems; Shapley value; cost sharing; job scheduling.
**JEL Classification:** C71, D63, C62, D71.

*We thank François Maniquet for several helpful discussions. We would like to thank François Maniquet, Hervé Moulin, Eilon Solan, and Rakesh Vohra for their comments on earlier drafts of this paper. We would also like to thank Anna Bogomolnaia, Sidartha Gordon, Jay Sethuraman, and the seminar participants at Center for Operations Research and Econometrics and Indian Institute of Sciences for their valuable feedback and criticism on this work. Few results from this work are due to appear as an extended abstract in the sixth ACM conference on Electronic Commerce [12].

# 1 Introduction

A set of jobs need to be served by a server which can process only one job at a time. Each job has a finite processing time and a per unit time waiting cost. Efficient ordering of this queue directs us to serve the jobs in decreasing order of the ratio of per unit time waiting cost and processing time, which is the *weighted shortest processing time* rule of Smith [18]. To compensate for waiting cost of jobs, monetary transfers to jobs are allowed. How should the jobs share the cost equitably amongst themselves (through transfers)?

The problem of fair division of costs among jobs in a queue is a natural setting to many applications. Such problems arise in various models of the Internet and manufacturing settings: computer programs are regularly scheduled on servers, data are scheduled to be transmitted over networks, and jobs are scheduled in shop-floor on machines. Queues appear in many public services, for instance in post offices, banks etc. Fair scheduling algorithms for various queueing models to route packets over networks has been studied extensively for over a decade by a community of Electrical Engineers (see [6]). In such applications with complicated queue set-ups they settle for fair schedules that compromise efficiency up to a certain threshold (this is termed Quality of Service). In all these applications there is a common resource (computer server, network bandwidth, shop-floor machine, teller in the bank) that is shared by a set of agents and the mode of service is through some queueing discipline. Moreover, there is no external force, such as the market, that determines the allocation of costs. Thus, the final allocation of costs is best decided through mutual agreement, or as dictated by the server respecting certain reasonable "*fairness*" criteria.

Determining the cost share of agents respecting fairness axioms has been a central problem in cooperative game theory. The general cost sharing literature is vast and has a long history. For a good survey on the theory of cost sharing, we refer to Moulin [15].

A paper by Maniquet [4] is the closest to our model and is the motivation behind our work. Maniquet [4] studies a model where he assumes all jobs have processing times equal to unity. For such a model, he characterizes the Shapley value rule using classical fairness axioms. Using a different definition of worth of coalitions, Chun [1] derives a "reverse" rule for the same model. In another paper, Chun [3] studies the envy properties of these rules. In the one dimensional models studied above, jobs are either identical (and hence every ordering is efficient) or not identical. In the two dimensional model there is a new level of heterogeneity with non-identical data where every ordering is efficient; we call these jobs of equal "priority" (ratio of per unit time waiting cost and processing time). To deal with the cost sharing between jobs of equal priority we find the axioms for the one dimensional models insufficient. We provide characterization of cost sharing in such a class of jobs using very intuitive set of axioms. Using this as the springboard, we are able to characterize the Shapley value rule for the general instances of non-identical jobs.

Independent of our work, Chun [2] has developed a characterization of the Shapley value for our model based on consistency and monotonicity axioms. But he assumes *binary trans-*

*fers*, i.e., buyers pay each other and the total amount a buyer receives is the sum of amounts he receives from other buyers. Our first set of characterization does not assume such binary transfers and is thus stronger in some sense. For our linear programming based characterization, we assume binary transfers. But, this characterization is based on a modified no-envy property. Moreover, we do not make use of consistency or monotonicity in any of our characterizations. Another recent work by Katta and Sethuraman [10], again independent of our work, characterizes the Shapley value solution using different sets of axioms (they do not assume binary transfers).

Another stream of literature is on "sequencing games", first introduced by Curiel et al. [5]. Curiel et al. [5] model is similar to ours, though their notion of worth of a coalition is very different from the one we consider here. They focus on sharing the *savings in costs* from a given initial ordering to the optimal ordering (also see Hamers et al. [9]). Recently, Klijn and Sánchez [11] considered a sequencing game without any initial ordering of jobs and show that such a game is balanced.

Strategic aspects of queueing problems have also been researched. Mitra [13] studies the *first best implementation* in queueing models with generic cost functions. First best implementation means that there exists an efficient mechanism in which jobs in the queue have a dominant strategy to reveal their true types and their transfers add up to zero. Suijs [19] shows that if waiting costs of jobs are linear then first best implementation is possible. Moulin [16] studies strategic concepts such as *splitting* and *merging* in queueing problems with equal per unit time waiting costs.

## 1.1 Our Contribution

We consider cost sharing (using the cooperative game theory approach) with general processing time and per unit time waiting costs. In cooperative game theory, the classical Shapley value rule is commonly applied to cost (surplus) sharing games. For instance, the Shapley value rule, when applied to the division of unproduced goods with monetary transfers and quasi-linear utility, satisfies many interesting fairness axioms (see Moulin [14]). Our main focus is the Shapley value rule and its axiomatic characterization in the setting of job scheduling problems.

In our problem, as a first attempt at cost sharing, we can consider two trivial cost sharing rules. In the first rule each job bears its own waiting cost as its cost share. In the other rule each job bears the waiting cost it inflicts on jobs behind it. In addition, each job bears its own processing cost in both the rules. It can be seen that the total cost (sans the processing cost) can be decomposed as the waiting cost of individual jobs or equivalently as the waiting cost inflicted by jobs on jobs behind them in the queue. In the first rule, the job placed first gets a huge discount while the last job has a large cost share. In the second rule the situation is reversed. The Shapley value rule simply suggests that we average these two cost shares. This clearly distributes the burden more evenly among the jobs (especially jobs in

the start and end of the queue).

We show that the Shapley value rule satisfies many intuitive fairness axioms. Firstly we introduce two axioms on how to share the costs when jobs have equal priority. The first axiom is on "merger" of jobs. It requires that if a set of jobs are merged such that the efficient ordering is unchanged and "externalities" (waiting cost incurred by a job and waiting cost inflicted to other jobs by a job) of a job remains the same, then the cost share of that job should remain the same after the merger. We call this the *independence of valid merging* (IVM) axiom. In our other important axiom, we consider two jobs of equal priority. In this case, in both the possible orderings, the second job incurs the same waiting cost due to the first job. In the *externality consistency* (EC) axiom, we require that in such a case if an allocation rule chooses two allocations with two different orderings, then the transfer of the job in the first (second) position should be the same in both the allocations. IVM axiom with EC axiom gives us the Shapley value for the equal priority case.

As an alternative to the IVM and EC axioms, we provide an axiom called *expected cost bound* (ECB) for the equal priority case. Since every ordering is efficient, if each ordering is equally likely to be chosen by the sever, then each job will have an expected waiting cost it inflicts on other jobs. ECB requires that in the the equal priority case, every job should pay its own processing cost and the expected cost it inflicts on other jobs, where the expectation is taken with respect to possible allocations (orderings). ECB axiom also gives us the Shapley value for the equal priority case.

Apart from these, we will require one of the following sets of axioms (that generalize corresponding axioms from Maniquet's work) to characterize the Shapley value rule. The independence axioms: cost share of a job is independent of preceding jobs' per unit time waiting cost and following jobs' processing time. The proportional responsibility axioms: the transfer to an additional job added to the end (beginning) of a queue is shared by the jobs before (after) it in proportion to their processing times (per unit time waiting costs). We characterize the Shapley value rule in three different ways using these axioms. In all the characterizations efficiency, Pareto indifference, and IVM with EC (or ECB) are imposed. Besides these, we either need the independence axioms or one of the proportional responsibility axioms in place of one of the independence axioms.

We then use linear programming to characterize the Shapley value rule. We observe that the relative ordering of jobs in an efficient ordering is also optimal to all the pairwise ordering problems (which can be written as linear programs). The corresponding dual solution for each of these pairwise problems can be interpreted as transfers between pairs of jobs. By reassembling these transfers (in a minimal way) we obtain the Shapley value transfers for all the jobs. Using the constraints posed by the dual optimal solutions (in the pairwise problems), we define the notion of *pairwise no-envy allocation*. A pairwise no-envy allocation is a transfer and an ordering pair such that for any pair of jobs, if those were the only two jobs in the system, they will not be better off changing the current ordering given the current transfers. We show that the Shapley value rule is the pairwise no-envy allocation for an

efficient ordering which minimizes sum of absolute values of transfers.

We also investigate rules which satisfy *no-envy*. In this regard, we show that *no-envy* is not possible, in general, in our model. Since the no-envy constraints are the same as competitive equilibrium constraints in our model, this implies that no competitive equilibrium exists in our model. However, in some special cases, it may be possible to achieve no-envy. We examine the special case where processing times of all the jobs are same. In this case, all rules satisfying no-envy are solutions to two linear programs which are dual to each other: (i) every efficient ordering is an optimal solution to the primal problem and (ii) the transfers are the corresponding dual optimal solutions. But these no-envy allocations need not be budget-balanced. We define the notion of "price of no-envy" in these setting and give an elegant method to compute it.

The rest of the paper is organized as follows. Section 2 describes the model and Section 3 discusses the Shapley value rule for the model. In Section 4, we discuss several fairness axioms. The characterization results involving fairness axioms appear in Section 5 and those involving pairwise no-envy allocations appear in Section 6. Section 7 discusses the issue of envy in our setting. We conclude with some discussion and a summary in Section 8.

# 2 The Model

There are $n$ jobs that need to be served by one server which can process only one job at a time. The set of jobs are denoted as $N := \{1, \ldots, n\}$. An ordering of the jobs is given by an one to one map $\sigma : N \to N$ and $\sigma_i$ denotes the position of job $i$ in that order. Given an ordering $\sigma$, define the followers of job $i$ by $F_i(\sigma) := \{j \in N : \sigma_i < \sigma_j\}$ and the predecessors of job $i$ by $P_i(\sigma) := \{j \in N : \sigma_i > \sigma_j\}$. We assume that for any $i \in N$ and any $\sigma$, if $F_i(\sigma)$ or $P_i(\sigma)$ is the empty set, then any summation over such sets gives zero.

Every job $i$ is identified by two parameters: $(p_i, \theta_i)$. $p_i$ is the processing time and $\theta_i$ is the per unit time waiting cost of job $i$. Thus, a *queueing problem* is defined by a list $q = (N, p, \theta) \in \mathbb{Q}$, where $\mathbb{Q}$ is the set of all possible lists. We will denote $\gamma_i = \frac{\theta_i}{p_i}$. We call $\gamma_i$, the *priority* of job $i$. Given an ordering of jobs $\sigma$, the cost incurred by job $i$ is given by

$$c_i(\sigma) = p_i\theta_i + \theta_i \sum_{j \in P_i(\sigma)} p_j.$$

The total cost incurred by all jobs due to an ordering $\sigma$ can be thought of in two ways: (i) by summing the cost incurred by every job and (ii) by summing the costs inflicted by a job on jobs behind it due to its own processing cost.

$$\begin{aligned} C(N, \sigma) = \sum_{i \in N} c_i(\sigma) &= \sum_{i \in N} p_i\theta_i + \sum_{i \in N} \Big[\theta_i \sum_{j \in P_i(\sigma)} p_j\Big]. \\ &= \sum_{i \in N} p_i\theta_i + \sum_{i \in N} \Big[p_i \sum_{j \in F_i(\sigma)} \theta_j\Big]. \end{aligned}$$

An *efficient ordering* $\sigma^*$ is the one which minimizes the total cost incurred by all jobs. So, $C(N, \sigma^*) \leq C(N, \sigma)\ \forall\ \sigma \in \Sigma$, where $\Sigma$ is the set of all orderings. For notational simplicity, we will write the total cost in an efficient ordering of jobs from $N$ as $C(N)$ whenever it is not confusing. Sometimes, we will deal only with a subset of jobs $S \subseteq N$. The ordering $\sigma$ will then be defined only on the jobs in $S$ and we will write $C(S)$ for the total cost from an efficient ordering of jobs in $S$. The following lemma shows that jobs are ordered in non-increasing priority in an efficient ordering. This is also known as the *weighted shortest processing time* rule, first introduced by Smith [18].

**Lemma 1** *For any $S \subseteq N$, let $\sigma^*$ be an efficient ordering of jobs in $S$. For every $i \neq j$, $i, j \in S$, if $\sigma^*_i > \sigma^*_j$, then $\gamma_i \leq \gamma_j$.*

*Proof*: Assume for contradiction that the statement of the lemma is not true. This means, we can find two consecutive jobs $i, j \in S$ ($\sigma^*_i = \sigma^*_j + 1$) such that $\gamma_i > \gamma_j$. Define a new ordering $\sigma$ by interchanging $i$ and $j$ in $\sigma^*$. The costs to jobs in $S \setminus \{i, j\}$ is not changed from $\sigma^*$ to $\sigma$. The difference between total costs in $\sigma^*$ and $\sigma$ is given by, $C(S, \sigma) - C(S, \sigma^*) = \theta_j p_i - \theta_i p_j$. From efficiency we get $\theta_j p_i - \theta_i p_j \geq 0$. This gives us $\gamma_j \geq \gamma_i$, which is a contradiction. ■

An *allocation* for $q = (N, p, \theta) \in \mathbb{Q}$ has two components: an ordering $\sigma$ and a transfer $t_i$ for every job $i \in N$. The payment received by job $i$ is denoted by $t_i$. Given a transfer $t_i$ and an ordering $\sigma$, the *cost share* of job $i$ is defined as,

$$\pi_i = c_i(\sigma) - t_i = \theta_i \sum_{j \in N : \sigma_j \leq \sigma_i} p_j - t_i.$$

An allocation $(\sigma, t)$ is *efficient* for $q = (N, p, \theta)$ whenever $\sigma$ is an efficient ordering and $\sum_{i \in N} t_i = 0$. $\sigma^*(q)$ will be used to denote an efficient ordering jobs in queue $q$ ($\sigma^*$ will be used when $q$ is understood from the context). The following straightforward lemma says that for two different efficient orderings, the cost share in one efficient allocation is possible to achieve in the other by appropriately modifying the transfers.

**Lemma 2** *Let $(\sigma, t)$ be an efficient allocation and $\pi$ be the vector of cost shares of jobs from this allocation. If $\sigma^* \neq \sigma$ is an efficient ordering and $t^*_i = c_i(\sigma^*) - \pi_i\ \forall\ i \in N$, then $(\sigma^*, t^*)$ is also an efficient allocation.*

*Proof*: Since $(\sigma, t)$ is efficient, $\sum_{i \in N} t_i = 0$. This gives $\sum_{i \in N} \pi_i = C(N)$. Since $\sigma^*$ is an efficient ordering, $\sum_{i \in N} c_i(\sigma^*) = C(N)$. This means, $\sum_{i \in N} t^*_i = \sum_{i \in N} [c_i(\sigma^*) - \pi_i] = 0$. So, $(\sigma^*, t^*)$ is an efficient allocation. ■

Depending on the transfers, the cost shares in different efficient allocations may differ. An *allocation rule* $\psi$ associates with every $q \in \mathbb{Q}$ a non-empty subset $\psi(q)$ of allocations.

# 3 Cost Sharing Using the Shapley Value

In this section, we define the coalitional cost of this game and analyze the solution given by the Shapley value. Given a queue $q \in \mathbb{Q}$, the cost of a coalition of $S \subseteq N$ jobs in the queue is defined as the cost incurred by jobs in $S$ if these are the only jobs served in the queue using an efficient ordering. Formally, the cost of a coalition $S \subseteq N$ is,

$$C(S) = \sum_{i \in S} \theta_i \sum_{j \in S : \sigma_j^* \leq \sigma_i^*} p_j,$$

where $\sigma^*$ $(= \sigma^*(S))$ is an efficient ordering considering jobs from $S$ only. The worth of a coalition of $S$ jobs is just $-C(S)$. This way of defining the worth of a coalition assumes that jobs in a coalition $S$ are served first and then the jobs not in the coalition $(N \setminus S)$ are served. Maniquet [4] observes another equivalent way to define the worth of a coalition is using the dual function of the cost function $C(\cdot)$. Another interesting way to define the worth of a coalition in such games is discussed by Chun [1], who assumes that the jobs in the coalition are served after the jobs not in the coalition are served.

Now, assume that the worth of a coalition is found with probability $\alpha$ $(0 \leq \alpha \leq 1)$ by our notion of defining the worth of a coalition and with probability $(1-\alpha)$ by Chun's notion of defining the worth of a coalition. In that case the worth of a coalition is given by

$$-C(S) = -\sum_{i \in S} \theta_i p_i - \alpha \sum_{i \in S} \theta_i \sum_{j \in S : \sigma_j(S) < \sigma_i(S)} p_j - (1-\alpha) \sum_{i \in S} \theta_i \sum_{j \in N \setminus S} p_j,$$

where $\sigma(S)$ is an efficient ordering of jobs in $S$. We will call this the *weighted coalition game* and derive the Shapley value of jobs from this notion of worth of a coalition.

The Shapley value (or cost share) of a job $i$ is defined as [17],

$$SV_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} \Big[ C(S \cup \{i\}) - C(S) \Big]. \tag{1}$$

The Shapley value rule says that jobs are ordered using an efficient ordering and transfers are assigned to jobs such that the cost share of job $i$ is given by Equation (1). It calculates the expected marginal contribution of a job to its predecessors where expectation is taken over all possible orderings. The expression in Equation (1) can be simplified further.

**Lemma 3** *Let $\sigma$ be an efficient ordering of jobs in the set $N$. For all $i \in N$ and $0 \leq \alpha \leq 1$ the Shapley value for the weighted coalition game is given by,*

$$SV_i = p_i\theta_i + \sum_{j : \sigma_j < \sigma_i} p_j\theta_i - \frac{1}{2} \Bigg[ \sum_{j : \sigma_j < \sigma_i} \Big[ \alpha p_j \theta_i + (1-\alpha) p_i \theta_j \Big] - \sum_{j : \sigma_j > \sigma_i} \Big[ \alpha p_i \theta_j + (1-\alpha) p_j \theta_i \Big] \Bigg].$$

The proof is given in the Appendix. By Lemma 3, the transfer corresponding to the Shapley value of the weighted coalition game is given by,

$$t_i = \frac{1}{2}\Bigg[\sum_{j:\sigma_j<\sigma_i}\Big[\alpha p_j\theta_i + (1-\alpha)p_i\theta_j\Big] - \sum_{j:\sigma_j>\sigma_i}\Big[\alpha p_i\theta_j + (1-\alpha)p_j\theta_i\Big]\Bigg], \tag{2}$$

where $\sigma$ is an efficient ordering. Also, observe the transfer values for two extreme values of $\alpha$. If $\alpha = 1$, we get $t_i = \frac{1}{2}\Big[\sum_{j:\sigma_j<\sigma_i} p_j\theta_i - \sum_{j:\sigma_j>\sigma_i} p_i\theta_j\Big]$ and if $\alpha = 0$, we get $t_i = \frac{1}{2}\Big[\sum_{j:\sigma_j<\sigma_i} p_i\theta_j - \sum_{j:\sigma_j>\sigma_i} p_j\theta_i\Big]$.

### 3.1 A Case for $\alpha = 1$

If we set $\alpha = 0$ in the weighted coalition game, there can be jobs which can have negative cost share. We give an example to illustrate this. Consider two jobs with $(p, \theta)$ values as: $(1, 5), (1, 1)$. When $\alpha = 0$, we get $C(\{1, 2\}) = 7, C(\{1\}) = 10, C(\{2\}) = 2$. Observe that $C(\{1\}) = 10 > 7 = C(\{1, 2\})$. The Shapley value for job 1 is $\frac{15}{2}$ and that of job 2 is $-\frac{1}{2}$. This shows that if we calculate the worth of a coalition by serving jobs not in the coalition first, then the resulting cost share from the Shapley value formula may be negative for some jobs. We feel this is not *fair* in many ways.

- In the example, job 2 does not bear even its own processing cost according to the Shapley value formula.
- In the example, the cost share of the second job decreases with the increase in per unit time waiting cost of the first job, whereas the the per unit time waiting cost of the first job does not influence the second job in any way.

So, for the rest of the paper, we will assume that worth of a coalition is defined by assuming jobs in the coalition are served first with probability 1 and then jobs not in the coalition are served. This means, **we will assume** $\alpha = 1$.

## 4 The Fairness Axioms

In this section, we will define several axioms on fairness and later characterize the Shapley value using them. For a given $q \in \mathbb{Q}$, we will denote $\psi(q)$ as the set of allocations from allocation rule $\psi$. Also, we will denote the cost share vector associated with an allocation rule $(\sigma, t)$ as $\pi$ and that with allocation rule $(\sigma', t')$ as $\pi'$ etc.

We will define three types of fairness axioms: (i) related to efficiency, (ii) related to equity, and (iii) related to independence.

## 4.1 Efficiency Axioms

We define two types of efficiency axioms. One related to efficiency which states that an efficient ordering should be selected and the transfers of jobs should add up to zero (*budget balance*).

**Definition 1** *An allocation rule $\psi$ satisfies* ***efficiency*** *if for every $q \in \mathbb{Q}$ and $(\sigma, t) \in \psi(q)$, $(\sigma, t)$ is an efficient allocation.*

The second axiom related to efficiency says that the allocation rule should not discriminate between two allocations which are equivalent to each other in terms of cost shares of jobs.

**Definition 2** *An allocation rule $\psi$ satisfies* ***Pareto indifference*** *if for every $q \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$, if there exists another allocation $(\sigma', t')$ such that $\Big[\pi_i = \pi'_i \; \forall \; i \in N\Big]$, then $(\sigma', t') \in \psi(q)$.*

An implication of Pareto indifference axiom and Lemma 2 is that if an allocation rule is nonempty then for every efficient ordering there is some set of transfers such that it is part of an efficient rule and the cost share of a job in all these allocations are the same. But Pareto indifference does not exclude the possibility of a job having different cost shares in two separate allocations of an allocation rule. Equal cost share in all allocations of an allocation rule can be made an axiom on its own. The Shapley value rule clearly satisfies such an axiom.

## 4.2 Equity Axioms

How should the cost be shared between two jobs if the jobs have some kind of similarity between them? Equity axioms provide us with fairness properties which help us answer this question. We provide several such axioms. Some of these axioms (for example anonymity, equal treatment of equals) are standard in the literature, while some are new.

We start with a well known equity axiom called anonymity. Denote $\rho : N \to N$ as a permutation of elements in $N$. Let $\rho(\sigma, t)$ denote the allocation obtained by permuting elements in $\sigma$ and $t$ according to $\rho$. Similarly, let $\rho(q)$ denote the new list of $(p, \theta)$ obtained by permuting elements of $p$ and $\theta$ according to $\rho$. Our first equity axiom states that allocation rules should be immune to such permutation of data.

**Definition 3** *An allocation rule $\psi$ satisfies* ***anonymity*** *if for all $q \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$ and every permutation $\rho$, then $\rho(\sigma, t) \in \psi(N, \rho(q))$.*

The next equity axiom is classical in literature and says that two jobs with equal parameters should be compensated such that their cost shares are also equal.

**Definition 4** *An allocation rule $\psi$ satisfies* ***equal treatment of equals (ETE)*** *if for all $q \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$, $i, j \in N$, then*

$$\Big[p_i = p_j; \theta_i = \theta_j\Big] \Rightarrow \Big[\pi_i = \pi_j\Big].$$

ETE directs us to share costs equally between jobs if they have the same per unit time waiting cost and processing time. At the same time it is silent about the cost shares of two jobs $i$ and $j$ that are indistinguishable with respect to an efficient ordering but have different parameters (with $\gamma_i = \gamma_j$). We introduce some new axioms towards resolving this lacuna.

We would like to introduce the idea of merging jobs with respect to job $i$. We would also like to focus on the case when all jobs are of equal priority. Suppose job $i$ is in position $\sigma_i$ in an ordering $\sigma$ of the queue. There are two costs that arise due to its existence in that position in the system. First is the waiting cost that job $i$ "feels" due to the processing times of jobs before it and second is the cost that jobs placed behind job $i$ feel due to the processing time of job $i$. When we consider the waiting cost, it is immaterial how job $i$ came to wait that length of time: whether it was due to a single job with large processing time or multiple jobs with smaller processing times. In the same vein, the cost job $i$ imposes on the jobs behind it depends only on the sum of their per unit time waiting costs and not on how these per unit time waiting costs were distributed among those jobs. Hence, as far as job $i$ is concerned we can merge all jobs before it with a processing time of $\sum_{j \in P_i(\sigma)} p_j$ and all jobs behind it with a per unit time waiting cost of $\sum_{j \in F_i(\sigma)} \theta_j$. By merging, we would like to think of these merged jobs as a single job with the above specified processing time (or per unit time waiting cost). However, to preserve the priority ($\gamma$) of jobs that we started out with we must set the per unit time waiting cost of the merged unit before as $\sum_{j \in P_i(\sigma)} \theta_j$ and processing time of the merged unit after as $\sum_{j \in F_i(\sigma)} p_j$. This means that the relative ordering remains intact; the jobs before (after) job $i$ that were merged can be placed before (after) job $i$. Since in the modified queue set up (with only three jobs) the "world view" of job $i$ with respect to its waiting cost or the cost it inflicts does not change, we would expect that it still preserves its cost share. This is the idea captured by our next axiom. We can generalize this idea of merging (with the same justification as above) to account for merging any subset of the jobs that are placed before or after $i$. We now present the technical definitions and details.

When any set of consecutive jobs $S \subseteq N$ are merged, they are treated like a single job with processing time $p_S = \sum_{i \in S} p_i$ and per unit time waiting cost $\theta_S = \sum_{i \in S} \theta_i$. We will denote the new (merged) job as $< S >$. Assume that we are given an efficient ordering $\sigma$ and a job $i \in N$. We will only consider mergers of consecutive jobs $S \subseteq F_i(\sigma)$ (or $T \subseteq P_i(\sigma)$). A merger $S$ (or $T$) is said to be a valid merger, if the new jobs are created by merging consecutive jobs and they have the parameters: $\sum_{j \in S} \theta_j$ and $\sum_{j \in S} p_j$ (or $\sum_{j \in T} \theta_j$ and $\sum_{j \in T} p_j$). A queue instance created by a particular choice of $S$ and $T$ ($S$ or $T$ can be $\emptyset$) is denoted by $q(S, T)$ and $M(\sigma, i)$ denotes the set of all such queue instances created using valid mergers. We recall here that (under the equal priority assumption) the choice of parameters for the new job ensures that $\gamma_i = \gamma_{<S>} = \gamma_{<T>}$ and hence the relative ordering still remains efficient.[1]

[1]Even if the jobs are not of equal priority, then also such merging of jobs results in an ordering that is efficient. In fact our valid merging axiom holds in the Shapley value rule for the general case when jobs are not of equal priority. But to characterize the Shapley value rule, we only need it to hold for the equal

**Definition 5** *An allocation rule $\psi$ satisfies* ***independence of valid merging (IVM)*** *if for all $q = (N, p, \theta) \in \mathbb{Q}$ with $\gamma_1 = \ldots, \gamma_n$, $(\sigma, t) \in \psi(q)$, $i \in N$, $q(S, T) \in M(\sigma, i)$, and $(\sigma', t') \in \psi(q(S, T))$, we have $\pi_i = \pi'_i$, where $\pi_i$ is the cost share of job $i$ in $(\sigma, t)$ and $\pi'_i$ is the cost share of job $i$ in $(\sigma', t')$.*

To motivate our next axiom, let us consider the case of two jobs with equal $\gamma$. There are only two possible orderings $\sigma$ with $\sigma_i = i$ for $i \in \{1, 2\}$ and the *reverse* ordering, denoted by $\sigma'$. In both the orderings the waiting cost of the second job is the same ($p_1\theta_2$ in $\sigma$ and $p_2\theta_1 = p_1\theta_2$ in $\sigma'$) and the first job does not incur any waiting cost. We assume that the jobs pay for their own processing cost and are concerned only with their "costs of interaction" (externality). Our next axiom requires that in this two job case when both orderings share the same cost distribution, any allocation rule should have the same transfer for the first (or the second) job in both the orderings.

**Definition 6** *An allocation rule $\psi$ satisfies* ***externality consistency (EC)*** *if for all $q = (N, p, \theta) \in \mathbb{Q}$ with $N = \{1, 2\}$ and $\gamma_1 = \gamma_2$, for any $(\sigma, t), (\sigma', t') \in \psi(q)$, we have $t_1 = t'_2$ and $t_2 = t'_1$.*

Of course, a rule may not choose both the allocations $(\sigma, t)$ and $(\sigma', t')$. But as we show in the next lemma under efficiency and Pareto indifference, it will choose these two allocations. The following Lemma characterizes the cost share of jobs when they have equal priority under efficiency, Pareto indifference, IVM, and EC.

**Lemma 4** *Consider $q \in \mathbb{Q}$ such that $\gamma_1 = \ldots = \gamma_n$. In an efficient allocation rule $\psi$ satisfying Pareto indifference, IVM, and EC, for every $i \in N$ the cost share of $i$ is $p_i\theta_i + \frac{1}{2}\theta_i \sum_{j \neq i} p_j = p_i\theta_i + \frac{1}{2}\Big[\theta_i \sum_{j \in P_i(\hat{\sigma})} p_j + p_i \sum_{j \in F_i(\hat{\sigma})} \theta_j\Big]$, where $\hat{\sigma}$ is any ordering of jobs in $N$.*

*Proof*: Let $(\sigma, t) \in \psi(q)$. Due to the equal $\gamma$ case, every ordering is efficient by Lemma 1. Consider a job $i \in N$ and any efficient ordering $\sigma'$ such that $\sigma'_i = 1$. By Lemma 2, there exists transfers $t'$ such that $c_i(\sigma) - t_i = c_i(\sigma') - t'_i$. By Pareto indifference, $(\sigma', t') \in \psi(q)$. Hence every rule will have a transfer vector to go with every efficient ordering.

Now, perform a valid merging of jobs in $F_i(\sigma')$ to form the new queue $q'$ with jobs $i$ and $< F_i(\sigma') >$. The equal $\gamma$ case is preserved by the valid merging as the new job $< F_i(\sigma') >$ has a processing time of $\sum_{j \neq i} p_j$ and per unit time waiting cost of $\sum_{j \neq i} \theta_j$ and $\gamma_i = \frac{\sum_{j \neq i} \theta_j}{\sum_{j \neq i} p_j}$. By IVM, the cost share of job $i$ does not change from $\psi(q)$ to $\psi(q')$. For simplicity we denote the job $< F_i(\sigma') >$ by $k$. Consider the two possible orderings $\sigma^1, \sigma^2$, where $\sigma^1_i = 1$ and $\sigma^2_i = 2$. By Pareto indifference, there exists two transfer vectors $t^1$ and $t^2$ such that $(\sigma^1, t^1), (\sigma^2, t^2) \in \psi(q')$. By EC, $t^1_i = t^2_k$ and $t^1_k = t^2_i$. Using $t^1_i + t^1_k = 0$, we get $t^1_i + t^2_i = 0$. By Pareto indifference, $c_i(\sigma^1) - t^1_i = c_i(\sigma^2) - t^2_i$. This gives, $t^1_i = \frac{1}{2}\Big[c_i(\sigma^1) - c_i(\sigma^2)\Big] = -\frac{1}{2}\theta_i \sum_{j \neq i} p_j$.

priority case.

This means, cost share of job $i$ is $p_i\theta_i + \frac{1}{2}\theta_i \sum_{j\neq i} p_j = p_i\theta_i + \frac{1}{2}\left[\theta_i \sum_{j\in P_i(\hat{\sigma})} p_j + p_i \sum_{j\in F_i(\hat{\sigma})} \theta_j\right]$, where $\hat{\sigma}$ is any ordering of jobs in $N$. This can be shown for every job in $N$. ■

Lemma 4 is the stepping stone to our axiomatic characterization results for the general two parameter case. It characterizes the costs shares of jobs for the equal priority case. Observe that in the model where all jobs have the same processing time (Maniquet's model [4]), the equal priority case reduces to the identical job case for which, by the ETE axiom, the total cost is shared equally among the jobs.

We present an alternate, but an intuitive, axiom to characterize the cost shares of jobs when $\gamma_1 = \ldots = \gamma_n$ and hence prove a lemma analogous to Lemma 4. An allocation rule $\psi$ chooses a non-empty set of allocations. For a queue instance $q \in \mathbb{Q}$, consider an allocation $(\sigma, t) \in \psi(q)$ chosen by allocation rule $\psi$. In the ordering $\sigma$, each job $i$ can be associated with the cost inflicted by it on another job $j$ (denoted by $\psi_{ij}(\sigma)$). $\psi_{ij}(\sigma) = p_i\theta_j$ if $\sigma_i < \sigma_j$, and 0 otherwise. For a job $i$, the expected cost it inflicts on other jobs is given by $\sum_{j\neq i} E(\psi_{ij})$, where $E(\psi_{ij})$ is the expected cost $i$ inflicts on $j$ in $\psi$ (taking expectation over the orderings chosen in $\psi$). Our next axiom says that every job should bear such expected inflicted cost and its own processing cost.

**Definition 7** *An allocation rule $\psi$ satisfies **expected cost bound (ECB)** if for all $q \in \mathbb{Q}$ with $\gamma_1 = \ldots = \gamma_n$, for every $i \in N$, for any $(\sigma, t) \in \psi(q)$, $\pi_i \geq p_i\theta_i + \sum_{j\neq i} E(\psi_{ij})$, where $\pi_i$ is the cost share of job $i$ in allocation $(\sigma, t)$.*

ECB gives a bound on the cost share of a job in the equal priority case. Such bounds on cost shares (utilities) are often imposed through individual rationality axioms in many cost sharing settings (see individual rationality axioms in Moulin [14] as an example). Using ECB, we can immediately obtain a lemma analogous to Lemma 4.

**Lemma 5** *Let $\gamma_1 = \ldots = \gamma_n$. In an efficient allocation rule $\psi$ satisfying Pareto indifference and ECB, for every $i \in N$, the cost share of $i$ is $\pi_i = p_i\theta_i + \frac{1}{2}p_i \sum_{j\neq i} \theta_j = p_i\theta_i + \frac{1}{2}\left[\theta_i \sum_{j\in P_i(\hat{\sigma})} p_j + p_i \sum_{j\in F_i(\hat{\sigma})} \theta_j\right]$, where $\hat{\sigma}$ is any ordering of jobs in $N$.*

*Proof*: Consider an allocation $(\sigma, t) \in \psi(q)$. Since all orderings are efficient and due to Pareto indifference, the probability of a job $i$ coming before another job $j$ is $\frac{1}{2}$ in $\sigma$. Thus the expected cost inflicted by $i$ on $j$ is $\frac{1}{2}p_i\theta_j$. Using ECB, we immediately get $\pi_i \geq p_i\theta_i + \frac{1}{2}p_i \sum_{j\neq i} \theta_j$. Assume for contradiction for some $i \in N$, $\pi_i > p_i\theta_i + \frac{1}{2}p_i \sum_{j\neq i} \theta_j$. So, we

get

$$\begin{aligned}
\sum_{i\in N}\pi_i &> \sum_{i\in N} p_i\theta_i + \frac{1}{2}\sum_{i\in N} p_i \sum_{j\neq i}\theta_j \\
&= \sum_{i\in N} p_i\theta_i + \frac{1}{2}\sum_{i\in N} p_i\Big[\sum_{j\in F_i(\sigma)}\theta_j + \sum_{j\in P_i(\sigma)}\theta_j\Big] \\
&= \sum_{i\in N} p_i\theta_i + \frac{1}{2}\sum_{i\in N} p_i \sum_{j\in F_i(\sigma)}\theta_j + \frac{1}{2}\sum_{i\in N}\theta_i \sum_{j\in P_i(\sigma)} p_j \quad \text{(Equal } \gamma \text{ case)} \\
&= p_i\theta_i + \sum_{i\in N} p_i \sum_{j\in F_i(\sigma)}\theta_j \quad \Big(\text{Using } \sum_{i\in N} p_i \sum_{j\in F_i(\sigma)}\theta_j = \sum_{i\in N}\theta_i \sum_{j\in P_i(\sigma)} p_j\Big) \\
&= C(N).
\end{aligned}$$

Imposing efficiency gives us a contradiction. So, $\pi_i = p_i\theta_i + \frac{1}{2}\theta_i\sum_{j\neq i} p_j = p_i\theta_i + \frac{1}{2}\Big[\theta_i\sum_{j\in P_i(\hat{\sigma})} p_j + p_i\sum_{j\in F_i(\hat{\sigma})}\theta_j\Big]$, where $\hat{\sigma}$ is any ordering of jobs in $N$. ■

We will revisit the equal priority case in Section 6.2, where we provide another characterization of cost shares from the Shapley value rule. We discuss two other ways, to approach this characterization in the Appendix.

Next, we introduce an axiom about sharing the transfer of a job between a set of jobs. In particular, if the last job quits the system, then the ordering need not change. But the transfer to the last job needs to be shared between the other jobs. This should be done in proportion to their processing times because every job influenced the last job based on its processing time.

**Definition 8** *An allocation rule $\psi$ satisfies* ***proportionate responsibility of*** $p$ ***(PR**$p$**)*** *if for all $q \in \mathbb{Q}$, for all $(\sigma, t) \in \psi(q)$, $k \in N$ such that $\sigma_k = |N|$, $q' = (N\setminus\{k\}, p', \theta') \in \mathbb{Q}$, such that for all $i \in N\setminus\{k\}$: $\theta'_i = \theta_i$, $p'_i = p_i$, there exists $(\sigma', t') \in \psi(q')$ such that for all $i \in N\setminus\{k\}$: $\sigma'_i = \sigma_i$ and*

$$t'_i = t_i + t_k\frac{p_i}{\sum_{j\neq k} p_j}.$$

An analogous fairness axiom results if we remove the job from the beginning of the queue. Since the presence of the first job influenced each job depending on their $\theta$ values, its transfer needs to be shared in proportion to $\theta$ values.

**Definition 9** *An allocation rule $\psi$ satisfies* ***proportionate responsibility of*** $\theta$ ***(PR**$\theta$**)*** *if for all $q \in \mathbb{Q}$, for all $(\sigma, t) \in \psi(q)$, $k \in N$ such that $\sigma_k = 1$, $q' = (N\setminus\{k\}, p', \theta') \in \mathbb{Q}$, such that for all $i \in N\setminus\{k\}$: $\theta'_i = \theta_i$, $p'_i = p_i$, there exists $(\sigma', t') \in \psi(q')$ such that for all $i \in N\setminus\{k\}$: $\sigma'_i = \sigma_i$ and*

$$t'_i = t_i + t_k\frac{\theta_i}{\sum_{j\neq k}\theta_j}.$$

The proportionate responsibility axioms are generalizations of *equal responsibility* axioms introduced by Maniquet [4].

### 4.3 Independence Axioms

The waiting cost of a job does not depend on the per unit time waiting cost of its preceding jobs. Similarly, the waiting cost inflicted by a job to its following jobs is independent of the processing times of the following jobs. These independence properties should hold for the cost sharing rules. This gives us the following two independence axioms.

**Definition 10** *An allocation rule $\psi$ satisfies* ***independence of preceding jobs' $\theta$ (IPJ$\theta$)*** *if for all $q = (N, p, \theta), q' = (N, p', \theta') \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$, $(\sigma', t') \in \psi(q')$, if for all $i \in N \setminus \{k\}$: $\theta_i = \theta'_i$, $p_i = p'_i$ and $\gamma_k < \gamma'_k$, $p_k = p'_k$, then for all $j \in N$ such that $\sigma_j > \sigma_k$: $\pi_j = \pi'_j$, where $\pi$ is the cost share in $(\sigma, t)$ and $\pi'$ is the cost share in $(\sigma', t')$.*

**Definition 11** *An allocation rule $\psi$ satisfies* ***independence of following jobs' $p$ (IFJ$p$)*** *if for all $q = (N, p, \theta), q' = (N, p', \theta') \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$, $(\sigma', t') \in \psi(q')$, if for all $i \in N \setminus \{k\}$: $\theta_i = \theta'_i$, $p_i = p'_i$ and $\gamma_k > \gamma'_k$, $\theta_k = \theta'_k$, then for all $j \in N$ such that $\sigma_j < \sigma_k$: $\pi_j = \pi'_j$, where $\pi$ is the cost share in $(\sigma, t)$ and $\pi'$ is the cost share in $(\sigma', t')$.*

## 5 The Characterization Results

In this section, we will see that all the fairness axioms discussed are satisfied by the Shapley value rule. Moreover, the Shapley value rule can be characterized by choosing appropriate subsets of these axioms. The next proposition shows that the Shapley value rule satisfies all the fairness axioms discussed.

**Proposition 1** *The Shapley value rule satisfies efficiency, Pareto indifference, anonymity, ETE, IVM, EC, ECB, IPJ$\theta$, IFJ$p$, PR$p$, and PR$\theta$.*

*Proof*: The Shapley value rule chooses an efficient ordering and by definition the payments add upto zero. So, it satisfies efficiency.

The Shapley value assigns same cost share to a job irrespective of the efficient ordering chosen. For every efficient ordering $\sigma$, we include all $(\sigma, t)$ in the Shapley value rule which gives the Shapley value cost shares to jobs. So, it is Pareto indifferent.

The Shapley value is anonymous because the particular index of a job does not effect its ordering or cost share.

For ETE, consider two jobs $i, j \in N$ such that $p_i = p_j$ and $\theta_i = \theta_j$. Without loss of generality assume the efficient ordering to be $1, \ldots, i, \ldots, j, \ldots, n$. Now, observe that,

$$\begin{aligned}\Big[L_j - L_i\Big] &= \Big[\sum_{k<j} p_k\theta_j - \sum_{k<i} p_k\theta_i\Big] \\ &= \sum_{i\le k<j} p_k\theta_j \text{ (Using } \theta_i = \theta_j) \\ &= \sum_{i<k\le j} p_j\theta_k \text{ (Using } \theta_i = \theta_j \text{ and } \gamma_k = \gamma_i \text{ for all } i \le k \le j) \\ &= \Big[R_i - R_j\Big] \text{ (Using } p_i = p_j).\end{aligned}$$

From Lemma 3 and $L_i + R_i = L_j + R_j$, we have

$$SV_i = p_i\theta_i + \frac{1}{2}\Big[L_i + R_i\Big] = p_j\theta_j + \frac{1}{2}\Big[L_j + R_j\Big] = SV_j.$$

Let $\gamma_1 = \ldots = \gamma_n$. Any ordering is efficient. If we do a valid merger of jobs before or after job $i$ in an efficient ordering, then the resulting new job will interact with job $i$ the same way as the set of jobs that was merged. The relative ordering of the jobs will also remain the same. This means the terms $\sum_{j\in P_i(\sigma)} p_j\theta_i$ and $\sum_{j\in F_i(\sigma)} p_i\theta_j$ will remain the same after a valid merger ($\sigma$ is an efficient ordering). Thus, the Shapley value for job $i$ remains the same. Hence, the Shapley value rule satisfies IVM.

For EC, consider two jobs: $(p_1, \theta_1), (p_2, \theta_2)$ of equal priority. The Shapley value of a job remains the same in whichever ordering you choose. This means, if job 1 is in the first position, its transfer is $-\frac{1}{2}p_1\theta_2 = -\frac{1}{2}p_2\theta_1$. But the transfer of job 1 in the first position is $-\frac{1}{2}p_2\theta_1$. A similar argument shows that the transfer of job 1 in the second position is equal to the transfer of job 2 in the second position. This means, the Shapley value rule satisfies EC.

In the equal priority case, the Shapley value of a job $i$ is $p_i\theta_i + \frac{1}{2}\sum_{j\neq i} p_i\theta_j = p_i\theta_i + \frac{1}{2}\sum_{j\neq i} E_{ij}(\psi)$, where $\psi$ is the Shapley value rule. So, the Shapley value rule satisfies ECB.

Consider any job $i$, in an efficient ordering $\sigma$, if we increase the value of $\gamma_j$ for some $j \neq i$ such that $\sigma_j > \sigma_i$, then the set $P_i$ (the set of preceding jobs) does not change in the new efficient ordering. If $\gamma_j$ is changed such that $p_j$ remains the same, then the expression $\sum_{j\in P_i} \theta_i p_j$ is unchanged. If $(p, \theta)$ values of no other jobs are changed, then the Shapley value is unchanged by increasing $\gamma_j$ for some $j \in P_i$ while keeping $p_j$ unchanged. Thus, the Shapley value rule satisfies IPJ$\theta$. An analogous argument shows that the the Shapley value rule satisfies IFJ$p$.

For PR$p$, assume without loss of generality that jobs are ordered $1, \ldots, n$ in an efficient ordering. Denote the transfer of job $i \neq n$ due to the Shapley value with set of jobs $N$ and

set of jobs $N \setminus \{n\}$ as $t_i$ and $t'_i$ respectively. Transfer of last job is $t_n = \frac{1}{2}\theta_n \sum_{j<n} p_j$. Now,

$$\begin{aligned} t_i &= \frac{1}{2}\Big[\theta_i \sum_{j<i} p_j - p_i \sum_{j>i} \theta_j\Big] \\ &= \frac{1}{2}\Big[\theta_i \sum_{j<i} p_j - p_i \sum_{j>i:j\neq n} \theta_j\Big] - \frac{1}{2} p_i \theta_n \\ &= t'_i - \frac{1}{2}\theta_n \sum_{j<n} p_j \frac{p_i}{\sum_{j<n} p_j} \\ &= t'_i - t_n \frac{p_i}{\sum_{j<n} p_j}. \end{aligned}$$

A similar argument shows that the Shapley value rule satisfies PR$\theta$. ■

We propose three different ways to characterize the Shapley value rule using our axioms. All our characterizations involve efficiency, Pareto indifference, IVM, and EC. Additionally we use IPJ$\theta$ with either of IFJ$p$ or PR$p$, or we use IFJ$p$ with either IPJ$\theta$ or PR$\theta$.

Our first characterization involves both the independence axioms with efficiency, Pareto indifference, IVM, and EC.

**Theorem 1** *An allocation rule $\psi$ satisfies efficiency, Pareto indifference, IVM, EC, IPJ$\theta$, and IFJ$p$ if and only if it is the Shapley value rule.*

*Proof*: The "if" part follows from Proposition 1.

Define for any $i, j \in N$, $\theta^i_j = \gamma_i p_j$ and $p^i_j = \frac{\theta_j}{\gamma_i}$. Assume without loss of generality that $\sigma$ is an efficient ordering with $\sigma_i = i \; \forall \; i \in N$ for $q = (N, p, \theta)$.

Consider the following $q' = (N, p', \theta')$ corresponding to job $i$ with $p'_j = p_j$ if $j \leq i$ and $p'_j = p^i_j$ if $j > i$, $\theta'_j = \theta^i_j$ if $j < i$ and $\theta'_j = \theta_j$ if $j \geq i$. Observe that all jobs have the same $\gamma$: $\gamma_i$ and thus, every ordering is efficient. By Lemma 2, Pareto indifference, and efficiency, $(\sigma, t') \in \psi(q')$ for some set of transfers $t'$. Using Lemma 4, we get cost share of $i$ from $(\sigma, t')$ as $\pi_i = p_i\theta_i + \frac{1}{2}\theta_i \sum_{j\neq i} p_j = p_i\theta_i + \frac{1}{2}\Big[L_i + R_i\Big]$. Now, for any $j < i$, if we change $\theta'_j$ to $\theta_j$ without changing processing time, the new $\gamma$ of $j$ is $\gamma_j \geq \gamma_i$. Applying IPJ$\theta$, the cost share of job $i$ should not change. Similarly, for any job $j > i$, if we change $p'_j$ to $p_j$ without changing $\theta_j$, the new $\gamma$ of $j$ is $\gamma_j \leq \gamma_i$. Applying IFJ$p$, the cost share of job $i$ should not change. Applying this procedure for every $j < i$ with IPJ$\theta$ and for every $j > i$ with IFJ$p$, we reach $q = (N, p, \theta)$ and the payoff of $i$ does not change from $\pi_i$. Using this argument for every $i \in N$ and using the expression for the Shapley value in Lemma 3, we get the Shapley value rule. ■

It is possible to replace one of the independence axioms with an equity axiom on sharing the transfer of a job. This is shown in Theorems 2 and 3.

**Theorem 2** *An allocation rule $\psi$ satisfies efficiency, Pareto indifference, IVM, EC, IPJ$\theta$, and PR$p$ if and only if it is the Shapley value rule.*

*Proof*: The "if" part follows from Proposition 1.

As in the proof of Theorem 1, define $\theta_j^i = \gamma_i p_j \ \forall \ i, j \in N$. Assume without loss of generality that $\sigma$ is an efficient ordering with $\sigma_i = i \ \forall \ i \in N$ for $q = (N, p, \theta)$.

Consider a queue with jobs in set $K = \{1, \ldots, i, i+1\}$, where $i < n$. Define $q' = (K, p, \theta')$, where $\theta'_j = \theta_j^{i+1} \ \forall \ j \in K$. Define $\sigma'_j = \sigma_j \ \forall \ j \in K$. $\sigma'$ is an efficient ordering for $q'$. By Lemma 2, Pareto indifference, and efficiency for some transfers $t'$ we have $(\sigma', t') \in \psi(q')$. By Lemma 4 the cost share of job $i+1$ in any allocation rule in $\psi$ must be $\pi_{i+1} = p_{i+1}\theta_{i+1} + \frac{1}{2}\Big[\sum_{j<i+1} p_j \theta_{i+1}\Big]$. Now, consider $q'' = (K, p, \theta'')$ such that $\theta''_j = \theta_j^i \ \forall \ j \leq i$ and $\theta''_{i+1} = \theta_{i+1}$. $\sigma'$ remains an efficient ordering in $q''$ and by IPJ$\theta$ the cost share of $i+1$ remains $\pi_{i+1}$. In $q''' = (K \setminus \{i+1\}, p, \theta'')$, we can calculate the cost share of job $i$ using Lemma 4 as $\pi_i = p_i\theta_i + \frac{1}{2}\sum_{j<i} p_j\theta_i$. So, using PR$p$ we get the new cost share of job $i$ in $q''$ as $\pi'_i = \pi_i + t_{i+1}\frac{p_i}{\sum_{j<i+1} p_j} = p_i\theta_i + \frac{1}{2}\Big[\sum_{j<i} p_j\theta_i + p_i\theta_{i+1}\Big]$.

Now, we can set $K = K \cup \{i+2\}$. As before, we can find cost share of $i+2$ in this queue as $\pi_{i+2} = p_{i+2}\theta_{i+2} + \frac{1}{2}\Big[\sum_{j<i+2} p_j\theta_{i+2}\Big]$. Using PR$p$ we get the new cost share of job $i$ in the new queue as $\pi_i = p_i\theta_i + \frac{1}{2}\Big[\sum_{j<i} p_j\theta_i + p_i\theta_{i+1} + p_i\theta_{i+2}\Big]$. This process can be repeated till we add job $n$ at which point cost share of $i$ is $p_i\theta_i + \frac{1}{2}\Big[\sum_{j<i} p_j\theta_i + \sum_{j>i} p_i\theta_j\Big]$. Then, we can adjust the $\theta$ of preceding jobs of $i$ to their original value and applying IPJ$\theta$, the payoffs of jobs $i$ through $n$ will not change. This gives us the Shapley values of jobs $i$ through $n$. Setting $i = 1$, we get cost shares of all the jobs from $\psi$ as the Shapley value. ■

**Theorem 3** *An allocation rule $\psi$ satisfies efficiency, Pareto indifference, IVM, EC, IFJp, and PR$\theta$ if and only if it is the Shapley value rule.*

*Proof*: The "if" part follows from Proposition 1.

The proof of "only if" part mirrors the proof of Theorem 2. We provide a short sketch. Analogous to the proof of Theorem 2, $\theta$s are kept equal to original data and processing times are initialized to $p_j^{i+1}$. This allows us to use IFJ$p$. Also, in contrast to Theorem 2, we consider $K = \{i, i+1, \ldots, n\}$ and repeatedly add jobs to the beginning of the queue maintaining the same efficient ordering. So, we add the cost components of preceding jobs to the cost share of jobs in each iteration and converge to the Shapley value rule. ■

Some comments about our characterization results and Maniquet's [4] characterization for the case of $p_1 = \ldots = p_n = 1$ are in order. Observe that we do not use the ETE axiom in our characterizations. But Maniquet uses the ETE axiom for his model. It is clear that identical jobs ought to have identical bargaining power (ETE axiom). But it is not clear as how bargaining power is distributed among jobs of equal priority. We have tried to establish this through IVM with EC and ECB (using invariance in ordering). In a sense, the ETE axiom in Maniquet's model makes the cost share of a job single-valued when every ordering of jobs is efficient. This cannot be achieved in our model using the ETE axiom. But it is

achieved using IVM with EC (Lemma 4) or ECB (Lemma 5) for our model. The "identical preferences lower bound" axiom used in [4] is not satisfied by the Shapley value rule in our model. So, no characterization is possible using it.

# 6 An Alternate Characterization

In the previous section, we stated some reasonable axioms (some standard and some new) and characterized the Shapley value rule using them. In this section, we take an alternate approach. The approach is motivated by the fact that the relative ordering of any pair of jobs in any efficient ordering gives an efficient ordering of that pair. Is it possible to look at transfers between these pairs of jobs and characterize the Shapley value rule for that ordering from that? In linear programming terms, the problem of finding an efficient ordering is a primal problem whose dual variables can be interpreted as transfers. In essence, we are assuming that jobs pay each other (in stead of being paid by the server) and the transfer between two jobs only depend on the costs they inflict on each other and is independent of the costs due to other jobs. This motivates our characterization in this section.

Consider two jobs $i$ and $j$ and the problem of ordering them. This can be done independent of other jobs present. In particular, we have the binary variables $x_{ik}$ (respectively $x_{jk}$) for $k = 1, 2$, which means that job $i$ (respectively $j$) is placed in position $k$. The problem of finding an efficient ordering between $i$ and $j$ can be written as a simple linear program using these variables.

$$C^{ij} = \min x_{i2} p_j \theta_i + x_{j2} p_i \theta_j$$

s.t. **(P(ij))**

$$\sum_{k=1,2} x_{ik} = 1 \tag{3}$$

$$\sum_{k=1,2} x_{jk} = 1 \tag{4}$$

$$x_{ik} + x_{jk} = 1 \text{ for } k = 1, 2$$

$$x_{ik}, x_{jk} \geq 0 \text{ for } k = 1, 2$$

The objective function minimizes the waiting cost due to the ordering of $i$ and $j$. We do not need to consider the waiting cost due to other jobs when we are ordering $i$ and $j$. The constraints are one-to-one assignment constraints.[2]

[2]It is well known that the feasible solution space of assignment problem has integral extreme points. Thus, we need not place binary constraints on the variables and a linear programming formulation gives us optimal solution to the problem.

The dual of formulation (**P(ij)**) gives us information about transfers.

$$C^{ij} = \max \pi_i^{ij} + \pi_j^{ij} + t_1^{ij} + t_2^{ij}.$$

s.t. (**D(ij)**)

$$\pi_i^{ij} + t_1^{ij} \leq 0 \tag{5}$$
$$\pi_j^{ij} + t_1^{ij} \leq 0 \tag{6}$$
$$\pi_i^{ij} + t_2^{ij} \leq p_j\theta_i \tag{7}$$
$$\pi_j^{ij} + t_2^{ij} \leq p_i\theta_j \tag{8}$$

The superscript $(ij)$ in the dual variables indicate that formulation is corresponding to the ordering of jobs $i$ and $j$. The relative ordering obtained from an efficient ordering to the original problem of $n$ jobs, is an optimal solution to formulation (**P(ij)**). Suppose that it is $x_{i1} = 1$ and $x_{j2} = 1$, then complementary slackness condition tells us that constraints (5) and (8) will be tight in the optimal solution of (**D(ij)**). This gives us the following set of equations:

$$\pi_i^{ij} + t_1^{ij} = 0 \tag{**CS-1**}$$
$$\pi_j^{ij} + t_2^{ij} = p_i\theta_j \tag{**CS-2**}$$

Substituting for $\pi_i^{ij}$ and $\pi_j^{ij}$ in constraints (6) and (7), we get that the following inequalities are satisfied at the optimal solution of (**D(ij)**).

$$t_2^{ij} - t_1^{ij} \leq p_j\theta_i \tag{9}$$
$$t_2^{ij} - t_1^{ij} \geq p_i\theta_j \tag{10}$$

If we interpret $t_1^{ij}$ to be the transfer of job in position 1 in the optimal solution of (**P(ij)**) (in this case job $i$) and $t_2^{ij}$ to be the transfer of job in position 2 in the optimal solution of (**D(ij)**) (in this case job $j$), then Equations (9) and (10) gives us a family of transfers. Since, we are considering only jobs $i$ and $j$, it is natural to impose the following *budget-balance* constraint on the transfers obtained in this manner.

$$t_1^{ij} + t_2^{ij} = 0 \tag{11}$$

Equation (11) says that the transfer made by job $i$ to job $j$ is equal to the transfer received by job $j$ from job $i$. Imposing Equation (11) on the dual optimal space, we get the following bound on the transfers.

$$t_1^{ij} = -t_2^{ij} \tag{12}$$
$$\frac{1}{2}p_j\theta_i \geq t_2^{ij} \geq \frac{1}{2}p_i\theta_j \tag{13}$$

Equations (12) and (13) further reduce the dual optimal solution space but still leaves us with a family of transfers. In the rest of section, we intend to show that one of these transfers corresponds to the Shapley value rule.

## 6.1 Pairwise No-Envy Transfers

Using the duality argument discussed, we are going to define some concepts. We will assume that jobs make transfers to each other. So, $t_{ij}$ is the transfer of job $i$ to job $j$ ($j \neq i$) and $t$ is the vector of such transfers. The total transfer of job $i$ is $\sum_{j \neq i} t_{ij}$. Implicit in this scheme is the assumption that $t_{ij} + t_{ji} = 0$. This means transfer made from $i$ to $j$ and $j$ to $i$ should add up to zero.

**Definition 12 (Pairwise No-Envy Allocation)** *An allocation $(\sigma, t)$ is a **pairwise no-envy allocation** if for every $i, j \in N$*

- $\sigma_i < \sigma_j \Rightarrow -t_{ij} \leq p_j\theta_i - t_{ji}$ *and* $p_i\theta_j - t_{ji} \leq -t_{ij}$.
- $t_{ij} + t_{ji} = 0$.

*The transfer $t$ is called a **pairwise no-envy transfer**.*

The conditions in the definition of pairwise no-envy allocation comes out of the optimal dual solution characterized earlier. A careful look at the first condition shows how a sense of no-envy comes out of this definition. If $\sigma_i < \sigma_j$, then the current cost share of job $i$ *due to* job $j$ is $-t_{ij}$ and that of job $j$ *due to* job $i$ is $p_i\theta_j - t_{ji}$. But if $i$ and $j$ switch positions, then the cost share of job $i$ *due to* job $j$ is $p_j\theta_i - t_{ji}$ and that of job $j$ *due to* job $i$ is $-t_{ij}$. The first condition says $i$ and $j$ are not better off switching their current positions. The second condition says that transfers between two jobs should add up to zero. In essence, the scenario has assumed no role for the server in calculating transfers. So, every pair of job forms a separate market to decide their transfers. We call such markets "markets of pairs", where two jobs decide their transfers based on the relative ordering between them imposed by the efficient ordering. The pairwise no-envy condition tells us that they determine these transfers purely based on their individual interaction and such that they would not prefer to switch their relative positions. This gives us a characterization of the Shapley value rule.

**Theorem 4** *Let $(\sigma, t)$ be a pairwise no-envy allocation. If $\sigma$ is an efficient ordering and $t$ minimizes $\sum_{i \in N} \sum_{j \neq i} |t_{ij}|$ over all possible transfers, then $(\sigma, t)$ is an allocation implied by the Shapley value rule.*

*Proof*: Consider any two jobs $i, j \in N$. Without loss of generality, assume that $\sigma_i < \sigma_j$. Pairwise no-envy allocation gives us $p_i\theta_j \leq t_{ji} - t_{ij} \leq p_j\theta_i$. Substituting, $t_{ij} + t_{ji} = 0$, we get $\frac{1}{2}p_i\theta_j \leq t_{ji} \leq \frac{1}{2}p_j\theta_i$. So, $t_{ji} > 0$ and $t_{ij} < 0$. Now, $\sum_{i \in N} \sum_{j \neq i} |t_{ij}| = \sum_{i \in N} \sum_{j:\sigma_i < \sigma_j} -t_{ij} + \sum_{i \in N} \sum_{j:\sigma_j < \sigma_i} t_{ij} = \sum_{i \in N} \sum_{j:\sigma_i < \sigma_j} (t_{ji} - t_{ij})$. Clearly, this is minimized if $t_{ji} = \frac{1}{2}p_i\theta_j$ and $t_{ij} = -\frac{1}{2}p_i\theta_j$ for every $i, j \in N$. This gives transfer of job $i$ as $\sum_{j \neq i} t_{ij} = \frac{1}{2}\Big[\sum_{\sigma_j < \sigma_i} p_j\theta_i - \sum_{\sigma_j > \sigma_i} p_i\theta_j\Big]$. So, cost share of $i$ in allocation $(\sigma, t)$ is $p_i\theta_i + \frac{1}{2}\Big[\sum_{\sigma_j < \sigma_i} p_j\theta_i + \sum_{\sigma_j > \sigma_i} p_i\theta_j\Big]$. By Lemma 3, this is the Shapley value cost share of job $i$. Since $\sigma$ is efficient, $(\sigma, t)$ is a rule implied by the Shapley value rule. ■

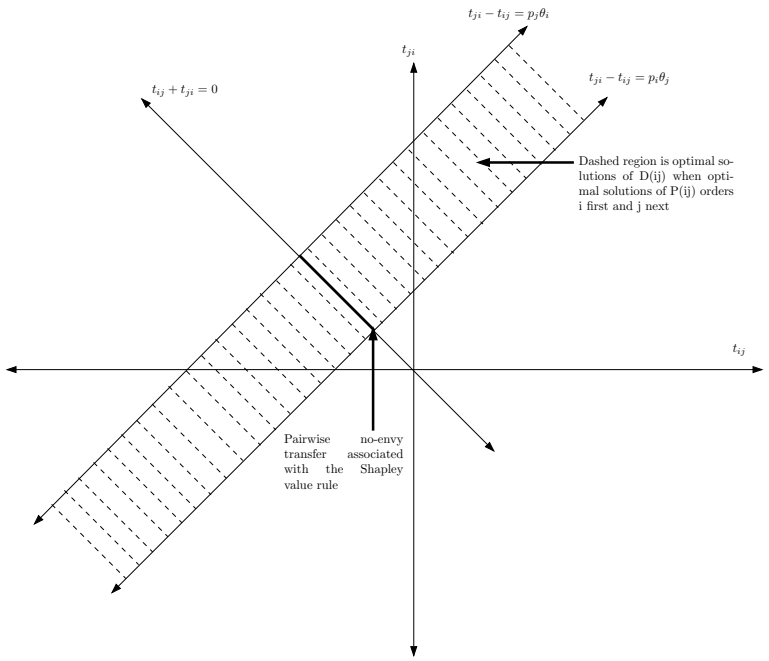


Figure 1: A Family of Transfers From Duality Arguments

Figure 1 gives a pictorial representation of the characterization. For every job pair $i, j \in N$, the first condition in an efficient pairwise no-envy allocation allows us to select transfers from the dashed region, corresponding to optimal solutions to the dual problem D(ij). By imposing the constraint $t_{ij} + t_{ji} = 0$, we get a smaller set of feasible transfers, denoted by the thick black line inside the dashed region. The lower extreme point of this line represents the point corresponding to the Shapley value rule. Adding the transfers from such extreme points from such figures corresponding to pairs of jobs in which $i$ is present gives us the Shapley value rule. This also minimizes the absolute value over all such possible transfers.

### 6.1.1 Family of Pairwise No-Envy Transfers

Using an argument similar to Theorem 4, we can show that an efficient and pairwise no-envy allocation $(\sigma, t)$ which maximizes the absolute value of transfers is a rule in which the transfer of job $i$ is

$$t_i = \frac{1}{2}\Big[\sum_{j:\sigma_i>\sigma_j} p_i\theta_j - \sum_{j:\sigma_i<\sigma_j} p_j\theta_i\Big].$$

This is the upper extreme point of the thick black line in Figure 1. From Equation (2), this is the transfer corresponding to the Shapley value when $\alpha = 0$ in the weighted coalition game (i.e., by defining the worth of a coalition by allowing jobs not in the coalition to be served first).

In the weighted coalition game, for any $0 \leq \alpha \leq 1$, we can get the transfers corresponding to the Shapley value by taking convex combination of the extreme points of the thick black line in Figure 1. Thus, the Shapley value rule corresponding to the weighted coalition game is completely characterized by the pairwise no-envy allocations (the thick black line in Figure 1).

### 6.1.2 Non-negative Cost Share Constraint

Non-negativity is a natural constraint to pose on the cost share of a job. In the 'market of pairs' setup, we can impose non-negativity of cost shares of jobs in every such market. This means, of all the optimal dual solution of (**D(ij)**), we are interested in those solutions in which $\pi_i^{ij} \geq 0$ and $\pi_j^{ij} \geq 0$. Assuming efficient (relative) ordering of $i$ and $j$ is $\sigma$ with $\sigma_i < \sigma_j$ and translating this in terms of transfers using Equations (**CS-1**) and (**CS-2**), we get $t_{ij} \leq 0$ and $t_{ji} \leq p_i\theta_j$ (replacing $t_1^{ij}$ by $t_{ij}$ and $t_2^{ij}$ by $t_{ji}$ in Equations (**CS-1**) and (**CS-2**)). Observe that these transfers make perfect sense (in terms of fairness): the earlier job should always pay the later job and the later job should not be compensated more than its waiting cost. Although the constraint $t_{ij} \leq 0$ is satisfied by all pairwise no-envy transfers, the constraint $t_{ji} \leq p_i\theta_j$ need not be satisfied by all the pairwise no-envy transfers. Specifically, if the value of $p_j\theta_i$ is very high, then there can be some pairwise no-envy transfers which violates this constraint. This means, there can be some pairwise no-envy transfers (specifically, corresponding to the upper extreme point of the thick black line in Figure 1) where cost share of a job may be negative. This is illustrated in Figure 2. The thick black line representing all the pairwise no-envy transfers has two distinct parts. The part which lies above the line $t_{ji} = p_i\theta_j$ corresponds to all the pairwise no-envy transfers which give negative cost share to job $j$ because of interaction between jobs $i$ and $j$. On the other hand, the part which lies below the line $t_{ji} = p_i\theta_j$ corresponds to all the pairwise no-envy transfers which give positive cost share to job $j$ because of interaction between jobs $i$ and $j$.

## 6.2 The Equal $\gamma$ Case

When $\gamma_1 = \ldots = \gamma_n$, the dual optimal region in Figure 1 is a straight line and it intersects the budget-balance line ($t_{ij} + t_{ji} = 0$) exactly at one point. This point corresponds to the Shapley value rule. Thus, in the equal $\gamma$ case, if we impose pairwise no-envy, we can reach at a lemma analogous to Lemma 4. This means that IVM with EC or ECB axioms can be replaced by pairwise no-envy allocation in equal $\gamma$ case in our characterizations in Theorems 1, 2, and 3.

# 7 No-envy in Cost Sharing

Chun [3] discusses a fairness condition called *no-envy* for the case when processing times of all jobs are unity. In general, no-envy, introduced by Foley [7], is a well-studied concept for

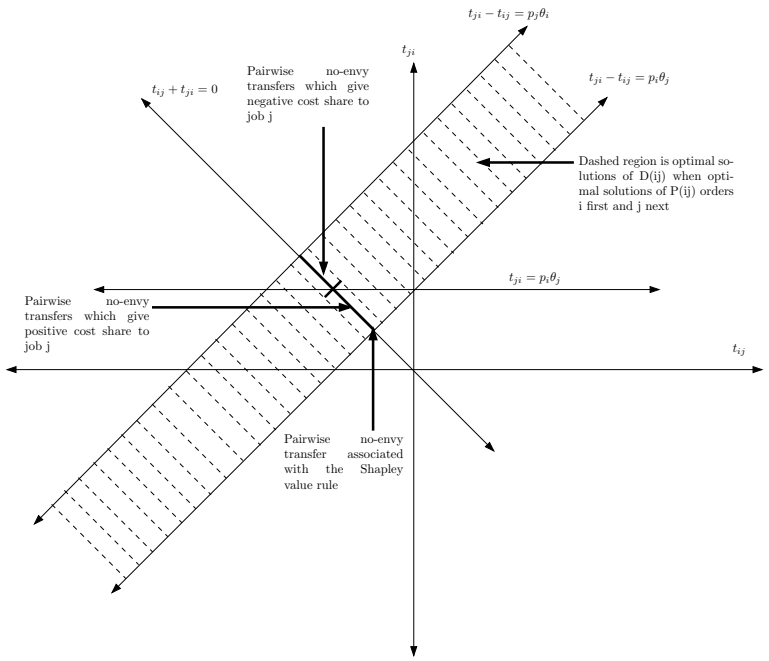


Figure 2: Pairwise No-envy Transfers May Give Negative Cost Share

a wide class of problems in economics. It requires that no job should end up with a higher payoff (in our case, a lower cost share) by consuming what any other job consumes (in our case being at a position of another job and receiving the transfer of that job). Formally, no-envy can be defined as follows.

**Definition 13** *An allocation rule satisfies* ***no-envy*** *if for all* $q \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$, *and for every* $i \in N$, *we have* $\pi_i \leq c_i(\sigma^{ij}) - t_j$ *for every* $j \neq i$, *where* $\pi_i$ *is the cost share of job* $i$ *from allocation rule* $(\sigma, t)$ *and* $\sigma^{ij}$ *is the ordering obtaining by swapping* $i$ *and* $j$ *in* $\sigma$.

From the result in [3], the Shapley value rule does not satisfy no-envy in our model also. To overcome this, Chun [3] introduces the notion of *adjusted no-envy*, which he shows is satisfied in the Shapley value rule when processing times of all jobs are unity. Here, we show that adjusted envy continues to hold for the Shapley value rule in our model (when processing times need not all be unity).

As before let $\sigma^{ij}$ denote an ordering where the position of $i$ and $j$ is swapped from an ordering $\sigma$. For adjusted no-envy, if $(\sigma, t)$ is an allocation for some $q \in \mathbb{Q}$, and let $t^{ij}$ be the transfer of job $i$ when the transfer of $i$ is calculated with respect to ordering $\sigma^{ij}$. Observe that an allocation may not allow for calculation of $t^{ij}$. For example, if $\psi$ is efficient, then $t^{ij}$ cannot be calculated if $\sigma^{ij}$ is not an efficient ordering. For simplicity, we state the definition of adjusted no-envy to apply to all such rules.

**Definition 14** *An allocation rule satisfies* ***adjusted no-envy*** *if for all* $q \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$, *and* $i, j \in N$, *we have* $\pi_i \leq c_i(\sigma^{ij}) - t_i^{ij}$.

**Proposition 2** *The Shapley value rule satisfies adjusted no-envy.*

*Proof*: Without loss of generality, assume efficient ordering of jobs is: $1, \ldots, n$. Consider two jobs $i$ and $i+k$. From Lemma 3,

$$SV_i = p_i\theta_i + \frac{1}{2}\Big[\sum_{j<i}\theta_i p_j + \sum_{j>i}\theta_j p_i\Big].$$

Let $\hat{\pi}_i$ be the cost share of $i$ due to adjusted transfer $t_i^{i(i+k)}$ in the ordering $\sigma^{i(i+k)}$.

$$\begin{aligned}
\hat{\pi}_i &= c_i(\sigma^{i(i+k)}) - t_i^{i(i+k)} \\
&= p_i\theta_i + \frac{1}{2}\Big[\sum_{j<i}\theta_i p_j + \theta_i p_{i+k} + \sum_{i<j<i+k}\theta_i p_j + \sum_{j>i}\theta_j p_i - \theta_{i+k}p_i - \sum_{i<j<i+k}\theta_j p_i\Big] \\
&= SV_i + \frac{1}{2}\sum_{i<j\leq i+k}\Big[\theta_i p_j - \theta_j p_i\Big] \\
&\geq SV_i \text{ (Using the fact that } \gamma_i \geq \gamma_j \text{ for } i<j).
\end{aligned}$$

■

Next, we ask what types of cost sharing rules satisfy no-envy. Our first observation in this regard is a representation of the no-envy constraints as a constraint in an appropriate directed graph. We say **no-envy is possible** in an allocation rule if there exists some transfers which satisfies no-envy.

Let $\sigma$ be the ordering chosen by an allocation rule. The no envy condition in Definition 13 can be written as

$$t_j - t_i \leq c_i(\sigma^{ij}) - c_i(\sigma) \;\forall\; i, j \in N. \tag{14}$$

Now, construct a complete directed graph (call it $Q$-graph) with nodes as the jobs and weights of $c_i(\sigma^{ij}) - c_i(\sigma)$ on the edge from job $j$ to job $i$. A famous result in graph theory says that the system of inequalities (14) has a feasible solution if and only if the resulting directed graph has no negative cycles. This shows no-envy is possible if and only if the $Q$-graph has no negative cycles.[3]

**Theorem 5** *No-envy is possible if and only if the $Q$-graph has no negative cycles. Further, the following statements are true.*

*(i) If the $Q$-graph has no negative cycles, then any rule which satisfies no-envy should choose an efficient ordering.*

*(ii) If $p_1 = \ldots = p_n$, then the $Q$-graph has no negative cycles.*

[3]This particular technique has been applied in a mechanism design context. Gui et al. [8] apply a similar graph theoretic technique to characterize dominant strategy mechanisms with quasi-linear utility and multi-dimensional types.

*Proof*: (i) We will show that if the $Q$-graph has no negative cycles, then the ordering $\sigma$ in an allocation rule satisfying no-envy is efficient. Without loss of generality, assume that $\sigma$ is the ordering defined by $1, \ldots, n$. If the $Q$-graph has no negative cycles, then considering the cycle between $i$ and $i+1$, we can write for any $i \in N$,

$$\begin{aligned} c_{i+1}(\sigma^{(i+1)i}) - c_{i+1}(\sigma) + c_i(\sigma^{i(i+1)}) - c_i(\sigma) \geq 0 \\ \Rightarrow -\theta_{i+1}p_i + \theta_i p_{i+1} \geq 0 \\ \Rightarrow \gamma_i \geq \gamma_{i+1}. \end{aligned}$$

This means (from Lemma 1) that $\sigma$ is an efficient ordering.

(ii) Consider the case $p_1 = p_2 = \ldots = p_n = p$. If $\sigma$ is an efficient ordering, then $\theta_1 \geq \theta_2 \ldots \geq \theta_n$. Now, without loss of generality consider the cycle, $i, j, k, \ldots, l, i$, where $i < j < k < l$. This means weight of the edge from job $i$ to job $j$ is $-\theta_j p(j-i)$. Similarly, weight of the edge from job $j$ to job $k$ is $-\theta_k p(k-j)$. Adding all weights of edges in the path from $i$ to $l$ we get the total weight of path from $i$ to $l$ as $\Delta_{il}$,

$$\Delta_{il} = -\theta_j p(j-i) - \theta_k p(k-j) - \ldots \tag{15}$$

Weight of edge defined from $l$ to $i$ is $\theta_i p(l-i)$. So, the total weight of cycle can be written as,

$$\begin{aligned} &\Delta_{il} + \theta_i(l-i) \\ &= -\theta_j p(j-i) - \theta_k p(k-j) - \ldots + \theta_i p(l-i) \\ &\geq -\theta_i p(j-i) - \theta_i p(k-j) - \ldots + \theta_i p(l-i) \\ &= 0. \end{aligned}$$

This means the $Q$-graph has no negative cycles. ■

In general, the $Q$-graph may have negative cycles, in which case no-envy is not possible. Consider the following example which shows the absence of no-envy. There are three jobs with the following parameters $(\theta, p)$: $(1,1), (2,3), (3,5)$. Assume for contradiction that no-envy rule exists. From Theorem 5, if no-envy rule exists it should select an efficient ordering. The efficient ordering in our case is: job 1 followed by job 2 followed by job 3. Let the transfer vector selected by the no-envy rule be $t$. Writing the no-envy rule for jobs 1 and 3, we get $-t_1 \leq 8 - t_3$ and $12 - t_3 \leq -t_1$. This implies $t_3 - t_1 \leq 8$ and $t_3 - t_1 \geq 12$, which is a contradiction. The intuition for the absence of no-envy is that the last job incurs too much waiting cost which needs to be compensated with high transfers (else it would like to switch to the first position). But the fist job does not incur so much waiting cost by switching to the last position. So, he prefers to switch to the last position to get the high transfer of the last job and low waiting cost.

We note that the no-envy constraints are competitive equilibrium constraints for our model. We can treat every position as an indivisible object which needs to be assigned to

jobs (agents). We can associate transfers with positions which are like prices. Value of a position for a job is negative of waiting cost the job incurs by switching position with the job in that position in an ordering. Thus, we arrive at an immediate corollary to Theorem 5

**Corollary 1** *The no-envy constraints are competitive equilibrium constraints and the absence of no-envy implies the absence of competitive equilibrium in our model.*

Also, note that since no-envy constraints are equivalent to competitive equilibrium constraints in our model, the first result in Theorem 5 is nothing but the first fundamental theorem of welfare economics.

## 7.1 The Case of Equal Processing Times

In the rest of the section, we will discuss a case when no-envy is possible and characterize these rules. We will focus on the case when $p_1 = \ldots = p_n = p$. From Theorem 5, no-envy is possible in this setting. Moreover, any rule which satisfies no-envy in such a setting should choose an efficient ordering.

We denote the set of positions in the system as $M = \{1, \ldots, j, \ldots, n\}$. When processing times are same ($p$) for all jobs, the waiting cost of job $i$ being served in position $j$ in the queue is $jp\theta_i$. We denote $c_{ij} = jp\theta_i$. In this case, the problem of finding an efficient ordering can be written as a linear program. Let $x_{ij}$ denote if job $i$ is assigned to position $j$.

$$C = \min \sum_{i \in N} \sum_{j \in M} c_{ij} x_{ij}$$

$$\text{s.t.} \qquad (\mathbf{P})$$

$$\sum_{j \in M} x_{ij} \geq 1 \qquad \forall\ i \in N$$

$$\sum_{i \in N} x_{ij} = 1 \qquad \forall\ j \in M$$

$$x_{ij} \geq 0 \qquad \forall\ i \in N,\ \forall\ j \in M.$$

This is the standard one-to-one assignment problem. The first set of constraints can be written as an equality. But it is written as an inequality so that the dual variables corresponding to that constraint can be interpreted nicely. The dual of formulation (**P**) can be written as follows.

$$C = \max \sum_{i \in N} \pi_i + \sum_{j \in M} t_j$$

$$\text{s.t.} \qquad (\mathbf{D})$$

$$\pi_i + t_j \leq c_{ij} \qquad \forall\ i \in N,\ \forall\ j \in M.$$

$$\pi_i \geq 0 \qquad \forall\ i \in N.$$

Observe that given $t$ variables at an optimal solution, we can find the $\pi$ variables by setting $\pi_i = \min_{j \in M} \left[ c_{ij} - t_j \right]$ (from complementary slackness). So, only $t$ variables are sufficient to describe an optimal solution of formulation (**D**). Without loss of generality, assume that the efficient ordering (optimal solution to (**P**)) assigns job $i$ to position $i$ for every $i \in N$. The non-negative dual variable $\pi_i$ can be interpreted as the cost share of job $i$ and the free variable $t_j$ can be interpreted as transfer of job at position $j$. In that case, the dual feasibility constraints say that for every job $i$ and position $j$, if $i$ goes to position $j$ and gets a transfer of $t_j$ (the transfer corresponding to position $j$), then the resulting cost share should be weakly worse off than $\pi_i$.

Let the transfers be associated with positions in stead of jobs. The following theorem says that formulations (**P**) and (**D**) characterizes rules satisfying no-envy.

**Theorem 6** *Let $p_1 = \ldots = p_n = p$ and $\psi$ be an allocation rule. $\psi$ satisfies no-envy if and only if for every $q \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$, $\sigma$ is an optimal solution of formulation (**P**) and $t$ is an optimal solution of formulation (**D**).*

*Proof*: For any $q \in \mathbb{Q}$, consider $(\sigma, t) \in \psi(q)$. Without loss of generality assume $\sigma_i = i$. If $\sigma$ is an optimal solution of (**P**) and $t$ is an optimal solution of (**D**), we can set $\pi_i = c_{ii} - t_i$ by complementary slackness. $\pi_i$ is nonnegative from feasibility of (**D**). Thus, $\pi_i$ represents the cost share of job $i$ in an efficient ordering with transfer $t_i$. The feasibility constraints in (**D**) are no-envy constraints. Thus, $\psi$ satisfies no-envy.

If $\psi$ satisfies no-envy, then for any $q \in \mathbb{Q}$, $(\sigma, t) \in \psi(q)$, $\sigma$ is an efficient ordering by Theorem 5. So, $\sigma$ is an optimal solution of (**P**). Without loss of generality assume that $\sigma_i = i$. Set $\pi_i = c_{ii} - t_i$. By no-envy $(\pi, t)$ is feasible. The complementary slackness condition requires that $\pi_i = c_{ii} - t_i$ for every $i \in N$, which is satisfied. By duality theory, a dual and primal feasible solution pair satisfying complementary slackness conditions are optimal solutions. So, $(\pi, t)$ is an optimal solution of (**D**). ■

Thus, solutions of (**P**) and (**D**) characterize completely the no-envy rules. Observe that a rule that satisfies no-envy need not be efficient. The missing constraint in the dual (**D**) is the budget-balance constraint in the efficient rule, $\sum_{j \in M} t_j = 0$. But in general, no-envy does not require transfers to add up to zero. In deed, an optimal solution of (**D**) may not exist such that $\sum_{j \in M} t_j = 0$. But if the system has only two jobs, there exists an efficient allocation which satisfies no-envy. In such a case, the primal and dual problem is the same as in pairwise no-envy allocation formulations (**P(ij)**) and (**D(ij)**). As shown in Figure 1, there exists a family of efficient allocations (corresponding to pairwise no-envy allocations) which satisfy no-envy. In fact, the Shapley value rule satisfies no-envy in this simple case.

### 7.1.1 The Price of No-Envy

As discussed, the no-envy solutions for the equal processing times case need not have budget balance. In that case, either the server may end up paying some transfers or receiving some

transfers on its own. This is the price the server (if $\sum_{j\in M} t_j > 0$) or the jobs (if $\sum_{j\in M} t_j < 0$) have to pay to achieve no-envy. We define $|\sum_{j\in M} t_j|$ as the **price of no-envy**. If we achieve budget-balance, then the price of no-envy is zero.

There is some no-envy solution for which the price of no-envy is minimum. There is a simple and elegant way to compute this minimum price of no-envy using linear programming. Let $C^*$ denote the objective function value of (**D**) in the optimal solution. Now, consider the following linear program.

$$\begin{aligned}
&\min \epsilon && \\
&\text{s.t.} && (\mathbf{D}^\epsilon)\\
&\sum_{i\in N}\pi_i + \sum_{j\in M} t_j = C^* && (16)\\
&\pi_i + t_j \le c_{ij} \qquad \forall\ i\in N,\ \forall\ j\in M && \\
&-\epsilon \le \sum_{j\in M} t_j \le \epsilon && \\
&\pi_i, \epsilon \ge 0 \qquad \forall\ i\in N. &&
\end{aligned}$$

Formulation ($\mathbf{D}^\epsilon$) finds a no-envy solution which is optimal by imposing the constraint (16). This is because any feasible solution $(\pi, t, \epsilon)$ of ($\mathbf{D}^\epsilon$) is such that $(\pi, t)$ is feasible in (**D**). Constraint (16) says that it has to be an optimal solution of (**D**). The $\epsilon$ variable at the optimal solution of ($\mathbf{D}^\epsilon$) gives us the minimum price of no-envy. This is achieved through the bounds placed on $\sum_{j\in M} t_j$ using the $\epsilon$ variable and minimizing the value of $\epsilon$.

# 8 Discussions and Summary

## 8.1 A Reasonable Class of Cost Sharing Mechanisms

In this section, we will define a *reasonable* class of cost sharing mechanisms. We will show how these reasonable mechanisms lead to the Shapley value mechanism. The following reasonable cost sharing mechanism requires that every job should be paid a constant fraction ($\alpha$) of its waiting cost and charged a constant fraction ($\beta$) of the cost it inflicts on other jobs.

**Definition 15** *An allocation rule $\psi$ is **reasonable** if for all $q \in \mathbb{Q}$ and $(\sigma, t) \in \psi(q)$ we have for all $i \in N$,*

$$t_i = \alpha\Big[\theta_i \sum_{j\in P_i(\sigma)} p_j\Big] - \beta\Big[p_i \sum_{j\in F_i(\sigma)} \theta_j\Big]\ \forall\ i\in N,$$

*where $0 \le \alpha \le 1$ and $0 \le \beta \le 1$.*

If we impose $\sum_{i\in N} t_i = 0$, we get $\alpha \sum_{i\in N} \theta_i \sum_{j\in P_i(\sigma)} p_j = \beta \sum_{i\in N} p_i \sum_{j\in F_i(\sigma)} \theta_j$. Consequently $\alpha = \beta$. If $\alpha = \beta = 0$, then every job bears its own cost. If $\alpha = \beta = 1$, then every job

gets compensated for its waiting cost but compensates others for the cost it inflicts on others. The Shapley value rule comes as a result of ETE as shown in the following proposition.

**Proposition 3** *Any efficient and reasonable allocation rule $\psi$ that satisfies ETE is a rule implied by the Shapley value rule.*

*Proof*: Consider a $q \in \mathbb{Q}$ in which $p_i = p_j$ and $\theta_i = \theta_j$. Let $(\sigma, t) \in \psi(q)$ and $\pi$ be the resulting cost shares. From ETE, we get,

$$\begin{aligned}
\pi_i &= \pi_j \\
\Rightarrow c_i(\sigma) - t_i &= c_j(\sigma) - t_j \\
\Rightarrow p_i\theta_i + (1-\alpha)L_i + \alpha R_i &= p_j\theta_j + (1-\alpha)L_j + \alpha R_j \text{ (Since } \psi \text{ is efficient and reasonable)} \\
\Rightarrow (1-\alpha)(L_i - L_j) &= \alpha(R_j - R_i) \text{ (Using } p_i = p_j, \theta_i = \theta_j) \\
\Rightarrow 1 - \alpha &= \alpha \text{ (Using } L_i - L_j = R_j - R_i \neq 0) \\
\Rightarrow \alpha &= \frac{1}{2}.
\end{aligned}$$

This gives us the Shapley value rule by Lemma 3. ■

## 8.2 Summary

We studied the problem of sharing costs for a job scheduling problem on a single server, when jobs have processing times and per unit time waiting costs. We took a cooperative game theory approach and showed that the famous Shapley value rule satisfies many nice fairness properties. We characterized the Shapley value rule using different intuitive fairness axioms. We also characterized the Shapley value rule using dual optimal solutions of a linear program which orders pairs of jobs in the queue efficiently. Finally, we looked at no-envy rules. We showed that in general there may not exist a no-envy rule. But when all jobs have the same processing time, the no-envy rules are characterized by a pair of linear programs which are dual to each other. Even in this special case, there may not exist a rule which is efficient and satisfies no-envy.

In future, we plan to further simplify some of the fairness axioms. We also plan to look at cost sharing mechanisms other than the Shapley value rule. Investigating the strategic power of jobs in such mechanisms is another line of future research that we are presently considering.

# References


[1] Youngsub Chun. A Note on Maniquet's Characterization of the Shapley Value in Queueing Problems. Working Paper, Rochester University, 2004.

[2] Youngsub Chun. Consistency and Monotonicity in Sequencing Problems. Working Paper, Seoul National University, 2004.

[3] Youngsub Chun. No-envy in Queueing Problems. Working Paper, Rochester University, 2004.

[4] François Maniquet. A Characterization of the Shapley Value in Queueing Problems. *Journal of Economic Theory*, 109:90–103, 2003.

[5] Imma Curiel, Giorgio Pederzoli, and Stef Tijs. Sequencing Games. *European Journal of Operational Research*, 40:344–351, 1989.

[6] Alan Demers, Srinivasan Keshav, and Scott Shenker. Analysis and Simulation of a Fair Queueing Algorithm. In *Proceedings of ACM SIGCOMM*, pages 1–12, August 1989.

[7] Duncan Foley. Resource Allocation and the Public Sector. *Yale Economic Essays*, 7:45–98, 1967.

[8] Hongwei Gui, Rudolf Müller, and Rakesh V. Vohra. Dominant Strategy Mechanisms with Multidimensional Types. Working Paper, Kellogg Graduate School of Management, Northwestern University, 2004.

[9] Herbert Hamers, Jeroen Suijs, Stef Tijs, and Peter Borm. The Split Core for Sequencing Games. *Games and Economic Behavior*, 15:165–176, 1996.

[10] Akshay-Kumar Katta and Jay Sethuraman. Cooperation in Queues. Working Paper, Columbia University, 2005.

[11] Flip Klijn and Estela Sánchez. Sequencing Games without Initial Order. Working Paper, Universitat Autónoma de Barcelona, July 2004.

[12] Debasis Mishra and Bharath Rangarajan. Cost Sharing in a Job Scheduling Problem Using the Shapley Value. In *Proceedings of sixth ACM Conference on Electronic Commerce (EC' 05)*, 2005.

[13] Manipushpak Mitra. Achieving the First Best in Sequencing Problems. *Review of Economic Design*, 7:75–91, 2002.

[14] Hervé Moulin. An Application of the Shapley Value to Fair Division with Money. *Econometrica*, 6(60):1331–1349, 1992.

[15] Hervé Moulin. *Handbook of Social Choice and Welfare*, chapter Axiomatic Cost and Surplus Sharing. North-Holland, 2002. Publishers: Arrow, Sen, Suzumura.

[16] Hervé Moulin. On Scheduling Fees to Prevent Merging, Splitting and Transferring of Jobs. Working Paper, Rice University, 2004.

[17] Lloyd S. Shapley. *Contributions to the Theory of Games II*, chapter A Value for $n$-person Games, pages 307–317. Annals of Mathematics Studies, 1953. Ediors: H. W. Kuhn, A. W. Tucker.

[18] Wayne E. Smith. Various Optimizers for Single-Stage Production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.

[19] Jeroen Suijs. On Incentive Compatibility and Budget Balancedness in Public Decision Making. *Economic Design*, 2:193–209, 2002.

# Appendix

## Proof of Lemma 3

*Proof*: The Shapley value for the weighted coalition rule can be written as $SV_i = \sum_{S \subseteq N: i \in S} \frac{\Delta(S)}{|S|}$, where $\Delta(S) = C(S)$ if $|S| = 1$ and $\Delta(S) = C(S) - \sum_{T \subsetneq S} \Delta(T)$. This means, $\Delta(\{i\}) = p_i\theta_i + (1-\alpha)\sum_{k \neq i} p_k\theta_i \; \forall \; i \in N$. For any $i, j \in N$ with $i \neq j$, we have

$$\begin{aligned}
\Delta(\{i,j\}) &= C(\{i,j\}) - C(\{i\}) - C(\{j\}) \\
&= p_i\theta_i + p_j\theta_j + \min(p_j\theta_i, p_i\theta_j) + (1-\alpha)\theta_i \sum_{k \notin \{i,j\}} p_k + (1-\alpha)\theta_j \sum_{k \notin \{i,j\}} p_k \\
&- p_i\theta_i - p_j\theta_j - (1-\alpha)\theta_i \sum_{k \neq i} p_k - (1-\alpha)\theta_j \sum_{k \neq j} p_k \\
&= \min(p_j\theta_i, p_i\theta_j) - (1-\alpha)(p_j\theta_i + p_i\theta_j).
\end{aligned}$$

By induction, we will show that $\Delta(S) = 0$ for $|S| > 2$. For $|S| = 3$, let $S = \{i, j, k\}$. Without loss of generality, assume $\gamma_i \geq \gamma_j \geq \gamma_k$. So, $\Delta(S) = C(S) - \Delta(\{i,j\}) - \Delta(\{j,k\}) - \Delta(\{i,k\}) - \Delta(\{i\}) - \Delta(\{j\}) - \Delta(\{k\}) = C(S) - p_i\theta_j + (1-\alpha)(p_j\theta_i + p_i\theta_j) - p_j\theta_k + (1-\alpha)(p_j\theta_k + p_k\theta_j) - p_i\theta_k + (1-\alpha)(p_i\theta_k + p_k\theta_i) - p_i\theta_i - (1-\alpha)\theta_i \sum_{l \neq i} p_l - p_j\theta_j - (1-\alpha)\theta_j \sum_{l \neq j} p_l - p_k\theta_k - (1-\alpha)\theta_k \sum_{l \neq k} p_l = C(S) - p_i\theta_i - p_j\theta_j - p_k\theta_k - p_i\theta_j - (p_i + p_j)\theta_k - (1-\alpha)\Big[\theta_i \sum_{l \neq i,j,k} p_l + \theta_j \sum_{l \neq i,j,k} p_l + \theta_k \sum_{l \neq i,j,k} p_l\Big] = C(S) - C(S) = 0$.

Now for $|S| > 3$, assume for $T \subsetneq S$, $\Delta(T) = 0$ if $|T| > 2$. Without loss of generality, assume $\sigma$ to be the identity mapping. Now,

$$\begin{aligned}
\Delta(S) &= C(S) - \sum_{T \subsetneq S} \Delta(T) \\
&= C(S) - \sum_{i \in S} \sum_{j \in S: j > i} \Delta(\{i,j\}) - \sum_{i \in S} \Delta(\{i\})
\end{aligned}$$

We will show that $\sum_{i \in S} \sum_{j \in S: j > i} \Delta(\{i,j\}) + \sum_{i \in S} \Delta(\{i\}) = C(S)$.

$$\begin{aligned}
\sum_{i \in S} \sum_{j \in S: j > i} \Delta(\{i,j\}) + \sum_{i \in S} \Delta(\{i\}) &= \sum_{i \in S} \sum_{j \in S: j > i} \Big[p_i\theta_j - (1-\alpha)(p_j\theta_i + p_i\theta_j)\Big] \\
&+ \sum_{i \in S} \Big[p_i\theta_i + (1-\alpha) \sum_{j \in N: j \neq i} p_j\theta_i\Big]
\end{aligned}$$

$$= \sum_{i\in S}\sum_{j\in S:j>i}\Big[\alpha p_i\theta_j - (1-\alpha)p_j\theta_i\Big] + \sum_{i\in S} p_i\theta_i$$
$$+ \sum_{i\in S}(1-\alpha)\Big[\sum_{j\in S:j>i} p_j\theta_i + \sum_{j\in S:j<i} p_j\theta_i + \sum_{j\in N\setminus S} p_j\theta_i\Big]$$

$$= \sum_{i\in S}\sum_{j\in S:j<i}\alpha p_j\theta_i - \sum_{i\in S}(1-\alpha)\sum_{j\in S:j>i} p_j\theta_i + \sum_{i\in S} p_i\theta_i$$
$$+ \sum_{i\in S}(1-\alpha)\Big[\sum_{j\in S:j>i} p_j\theta_i + \sum_{j\in S:j<i} p_j\theta_i + \sum_{j\in N\setminus S} p_j\theta_i\Big]$$
$$\text{(Using } \sum_{i\in S}\sum_{j\in S:j>i} p_i\theta_j = \sum_{i\in S}\sum_{j\in S:j<i} p_j\theta_i\text{)}$$
$$= \sum_{i\in S} p_i\theta_i + \sum_{i\in S}\sum_{j\in S:j<i} p_j\theta_i + \sum_{i\in S}(1-\alpha)\sum_{j\in N\setminus S} p_j\theta_i = C(S).$$

By induction, $\Delta(S) = 0$ for $|S| > 2$. Using the Shapley value formula for the weighted coalition rule,

$$SV_i = \sum_{S\subseteq N:i\in S}\frac{\Delta(S)}{|S|} = \Delta(\{i\}) + \frac{1}{2}\sum_{j\in N:j\neq i}\Delta(\{i,j\})$$
$$= p_i\theta_i + (1-\alpha)\theta_i\sum_{j\neq i} p_j$$
$$+ \frac{1}{2}\Bigg[\sum_{j\in N:j<i}\Big[p_j\theta_i - (1-\alpha)(p_j\theta_i + p_i\theta_j)\Big] + \sum_{j\in N:j>i}\Big[p_i\theta_j - (1-\alpha)(p_j\theta_i + p_i\theta_j)\Big]\Bigg]$$
$$= p_i\theta_i + \sum_{j\in N:j<i} p_j\theta_i - \frac{1}{2}\Bigg[\sum_{j\in N:j<i}\Big[\alpha p_j\theta_i + (1-\alpha)p_i\theta_j\Big] - \sum_{j\in N:j>i}\Big[\alpha p_i\theta_j + (1-\alpha)p_j\theta_i\Big]\Bigg].$$

■

## A Symmetry Axiom

In this section, we will derive Lemma 4 using a symmetry axiom. Thus, we answer the question of how to share the costs of jobs in the equal $\gamma$ case using a different axiom.

Every job inflicts some cost to the system because of its presence. It has two components: (i) its own processing cost and (ii) waiting cost it inflicts on its followers. We call this the *inflicted cost*. Denote the inflicted cost of job $i$ in an ordering $\sigma$ as $\hat{c}_i(\sigma)$. As before, we will continue to denote $c_i(\sigma)$ as the total cost incurred by job $i$ in an ordering $\sigma$.

Our current approach designs rules which redistributes the costs incurred by jobs. An alternate approach will be to distribute inflicted costs. For a rule $\psi$ and any queue instance, if

$(\sigma, t)$ is the allocation chosen, then $\kappa_i(\sigma, t) = \hat{c}_i(\sigma) + t_i$ gives the total cost share of the *system* for job $i$. This is different from cost share of job $i$, which is given as $\pi_i(\sigma, t) = c_i(\sigma) - t_i$. But given an allocation $(\sigma, t)$, we can distribute costs in terms of cost share of jobs $(\pi)$ or cost share of system $(\kappa)$. Both are equivalent ways to share the total cost of the system. The first method shares waiting costs while the latter method shares inflicted costs. We will impose an axiom which will require that any *symmetry* in inflicted cost and cost incurred should also be translated to respective cost sharing methods. Now, consider the following definition.

**Definition 16** *Two orderings $\sigma$ and $\sigma'$ are **symmetric** of each other with respect to job $i$, iff $c_i(\sigma) = \hat{c}_i(\sigma')$ and $\hat{c}_i(\sigma) = c_i(\sigma')$. Two allocations $(\sigma, t)$ and $(\sigma', t')$ are **symmetric** of each other with respect to job $i$ iff $\pi_i(\sigma, t) = \kappa_i(\sigma', t')$ and $\pi_i(\sigma', t') = \kappa_i(\sigma, t)$.*

Our next axiom says that if a rule chooses two allocations such that the orderings are symmetric of each other with respect every job, then the allocations should also be symmetric of each other with respect to every job.

**Definition 17** *An allocation rule $\psi$ satisfies **symmetry** if for all $q \in \mathbb{Q}$ and any $(\sigma, t), (\sigma', t') \in \psi(q)$, if $\sigma$ and $\sigma'$ are symmetric with respect to every job in $N$, then $(\sigma, t)$ and $(\sigma', t')$ are also symmetric with respect to every job in $N$.*

Let us consider the case when $\gamma_1 = \ldots = \gamma_n$. We call this the *equal* $\gamma$ case. The symmetry axiom tells us how to share costs in the equal $\gamma$ case as shown in the next Lemma.

**Lemma 6** *Consider a $q \in \mathbb{Q}$ such that $\gamma_1 = \ldots = \gamma_n$. In an efficient allocation rule $\psi$ satisfying Pareto indifference and symmetry, for every job $i \in N$, the cost share of $i$ is given as $p_i\theta_i + \frac{1}{2}p_i \sum_{j \neq i} \theta_j = p_i\theta_i + \frac{1}{2}\Big[\theta_i \sum_{j \in P_i(\hat{\sigma})} p_j + p_i \sum_{j \in F_i(\hat{\sigma})} \theta_j\Big]$, where $\hat{\sigma}$ is any ordering of jobs in $N$.*

*Proof*: Let $(\sigma, t) \in \psi(q)$. $\sigma$ is an efficient ordering. Construct $\sigma'$ by reversing the ordering $\sigma$. Due to the equal $\gamma$ case, $\sigma'$ is an efficient ordering. From Lemma 2 there exists transfers $t'$ such that $(\sigma', t')$ is an efficient allocation and $c_i(\sigma) - t_i = c_i(\sigma') - t'_i$ for all $i \in N$. Since $\psi$ satisfies Pareto indifference $(\sigma', t') \in \psi(q)$.

In the equal $\gamma$ case, $p_i\theta_j = \theta_i p_j$ for any pair of jobs $i, j \in N$ and $i \neq j$. Consider any job $i \in N$. $c_i(\sigma) = p_i\theta_i + \theta_i \sum_{j \in P_i(\sigma)} p_j = p_i\theta_i + p_i \sum_{j \in P_i(\sigma)} \theta_j = p_i\theta_i + p_i \sum_{j \in F_i(\sigma')} \theta_j = \hat{c}_i(\sigma')$. Similarly, $\hat{c}_i(\sigma) = c_i(\sigma')$. This means $\sigma$ and $\sigma'$ are symmetric with respect to every job in $N$. Since, $\psi$ satisfies symmetry, $(\sigma, t)$ and $(\sigma', t')$ should also be symmetric with respect to every job in $N$.

This means for every job $i \in N$, $\pi_i(\sigma, t) = \kappa_i(\sigma', t')$ and $\pi_i(\sigma', t') = \kappa_i(\sigma, t)$. Due to Pareto indifference $\pi_i(\sigma, t) = \pi_i(\sigma', t')$, which implies $\pi_i(\sigma, t) = \kappa_i(\sigma, t)$. Simplifying, $c_i(\sigma) - t_i = \hat{c}_i(\sigma) + t_i$. This gives, $t_i = \frac{1}{2}\Big[c_i(\sigma) - \hat{c}_i(\sigma)\Big]$. This means the cost share of job $i$ is $c_i(\sigma) - t_i = \frac{1}{2}\Big[c_i(\sigma) + \hat{c}_i(\sigma)\Big] = p_i\theta_i + \frac{1}{2}p_i \sum_{j \neq i} \theta_j = p_i\theta_i + \frac{1}{2}\Big[\theta_i \sum_{j \in P_i(\hat{\sigma})} p_j + p_i \sum_{j \in F_i(\hat{\sigma})} \theta_j\Big]$, where $\hat{\sigma}$ is any ordering of jobs in $N$. This can be shown for every job $i \in N$. ■

## Zero-Sum Extreme Transfers

In this section, we will derive Lemma 4 using a different axiom. This axiom is based on transfers of jobs at the beginning and end of a queue when all jobs have equal priority.

Let us consider the case when $\gamma_1 = \ldots = \gamma_n$. Observe that by Lemma 1, every ordering is efficient in the equal priority case. If a job is placed at the beginning of such a queue, it inflicts waiting cost (equal to $p_i \sum_{j \neq i} \theta_j$) to other jobs. This waiting cost exactly equals the waiting cost it would incur ($\theta_i \sum_{j \neq i} p_j = p_i \sum_{j \neq i} \theta_j$) if it was at the end of the queue. It incurs zero cost being at the first position and inflicts zero waiting cost on other jobs being at the last position. Our axiom in this section demands that such contrast in externality translates to transfers in the same manner. This means, if every ordering is efficient, then the amount a job pays at the beginning of the queue should equal the amount it would have received if it were at the end of the queue.

**Definition 18** *An allocation rule $\psi$ satisfies **zero-sum extreme transfers (ZET)** if for all $q \in \mathbb{Q}$ with $\gamma_1 = \ldots = \gamma_n$ and any $(\sigma, t) \in \psi(q)$ and $(\sigma', t') \in \psi(q)$, such that $\sigma_i = 1$ and $\sigma'_i = n$, then $t_i + t'_i = 0$.*

In general, an allocation rule may not choose two allocations as specified in ZET. But if a rule satisfies efficiency and Pareto indifference, then it will choose two allocations as specified in the ZET axiom. This gives us a lemma analogous to Lemma 4.

**Lemma 7** *Let $\gamma_1 = \ldots = \gamma_n$. In an efficient allocation rule $\psi$ satisfying Pareto indifference and ZET, for every $i \in N$, the cost share of $i$ is equal to $p_i\theta_i + \frac{1}{2}\theta_i \sum_{j \neq i} p_j = p_i\theta_i + \frac{1}{2}\Big[\theta_i \sum_{j \in P_i(\sigma)} p_j + p_i \sum_{j \in F_i(\sigma)} \theta_j$, where $\sigma$ is any ordering of jobs in $N$.*

*Proof*: Let $(\sigma, t) \in \psi(q)$. $\sigma$ is an efficient ordering. Let $\sigma'$ be an efficient ordering. From Lemma 2 there exists transfers $t'$ such that $(\sigma', t')$ is an efficient allocation and $c_i(\sigma) - t_i = c_i(\sigma') - t'_i$. Since $\psi$ satisfies Pareto indifference $(\sigma', t') \in \psi(q)$.

Consider any job $i$. There exists an efficient ordering $\sigma$ in which $\sigma_i = 1$. As argued before for some transfers $t$, we have $(\sigma, t) \in \psi(q)$. There exists an efficient ordering $\sigma'$ in which $\sigma'_i = n$. Again, there exists transfers $t'$ such that $(\sigma', t') \in \psi(q)$ and,

$$\begin{aligned} c_i(\sigma) - t_i &= c_i(\sigma') - t'_i \\ \Rightarrow p_i\theta_i - t_i &= p_i\theta_i + \theta_i \sum_{j \neq i} p_j - t'_i \\ \Rightarrow t_i &= -\frac{1}{2}\theta_i \sum_{j \neq i} p_j \text{ (By ZET).} \end{aligned}$$

This means the cost share of job $i$ in any allocation $(\sigma, t) \in \psi(q)$ is equal to $p_i\theta_i + \frac{1}{2}\theta_i \sum_{j \neq i} p_j = p_i\theta_i + \frac{1}{2}\Big[\theta_i \sum_{j \in P_i(\sigma)} p_j + p_i \sum_{j \in F_i(\sigma)} \theta_j$, where $\sigma$ is any ordering of jobs in $N$. This can be shown for every job $i \in N$. ■