A New Fairness Model Based on User's Objective for Multi-user Multi-processor Online Scheduling Problem

Debasis Dwibedy and Rakesh Mohanty

Abstract In multi-user multi-processor online scheduling, resources are shared among competing users, and fairness is considered to be a major performance criterion for resource allocation by the scheduler. Fairness ensures equality of resource sharing among the users. According to our knowledge, fairness based on the user's objective has neither been thoroughly explored nor a formal model has been well-defined in the literature. In this article, we propose a new fairness model for *Multi-user Multi-processor Online Scheduling Problem (MUMPOSP)*. We introduce and formally define quantitative fairness measures for an online scheduling algorithm based on optimization of makespan as an user's objective. Furthermore, we define unfairness and absolute fairness for an online scheduling algorithm. Lower bound results are shown for absolute fairness in a scheduling framework of equal length jobs. We show that our proposed fairness model can also measure algorithmic fairness by considering well-known optimality criteria such as sum of completion times, weighted sum of completion times and sum of flow times.

Keywords Multi-user system · Scheduling · Makespan · Performance measure · Fairness

1 Introduction

Fairness is an important performance criterion for a scheduler. Particularly, in multi-user systems, where several users compete for a set of resources (e.g., processor, memory) in order to achieve their objectives, the scheduler must guarantee fairness with respect to allocation of resources and user's objective. Though fairness has been studied based on resource allocation policies in the literature, there is less attention to devise a quantitative well-defined measure of fairness based on user's objectives.

User's objective as a fairness parameter has been motivated from the prevalent use of Web servers in client–server networking, grids and clusters in high-performance computing (HPC). Edge nodes in edge computing, service-oriented systems (SoS)

D. Dwibedy (✉) · R. Mohanty
Veer Surendra Sai University of Technology, Burla 768018, India

R. Patgiri et al. (eds.), *Edge Analytics*, Lecture Notes in Electrical Engineering 869,
https://doi.org/10.1007/978-981-19-0019-8_34

and supercomputers [1]. Unlike the traditional computing systems such as personal computer, the *SoS* supports multiple users. The users compete for system's resources for execution of their respective jobs. The most popular cluster scheduler *MAUI* [2] and the well-known *BOINC* platform [3] deal with a number of competing users, where each user submits a set of jobs simultaneously and desires *minimum time of completion (makespan)* for its submissions. A non-trivial challenge for the scheduler is to schedule jobs of multiple users in such a way that each user obtains its desired makespan.

Multi-user Multi-processor Online Scheduling Problem (MUMPOSP)

- **Inputs**: We are given a set $M = \{M_1, M_2, \ldots, M_m\}$ of m identical processors and a set of n jobs, where $m \geq 2$ and $n >>> m$. Let U_r represents a *user*, where $1 \leq r \leq k$ and J^r is the sequence of jobs requested by *user* U_r, where $J^r = (J_i^r | 1 \leq i \leq n_r)$ such that $J = \bigcup_{r=1}^{k} J^r$, $\sum_{r=1}^{k} n_r = n$ and $J^x \cap J^y = \phi$, where $x \neq y$ and $1 \leq x, y \leq k$. The processing time of job J_i^r is p_i^r, where $p_i^r \geq 1$.
- **Output**: A schedule (S) in which makespan for each U_r is denoted by $C_{\max}^r = \max\{c_i^r | 1 \leq i \leq n_r\}$, where c_i^r is the completion time of job J_i^r
- **Objective**: Minimization of $C_{\max}^r$, $\forall U_r$.
- **Constraint**: The scheduler can receive a batch of at most r jobs at any time step, and the jobs must be irrevocably scheduled before the arrival of next batch of jobs, where $1 \leq r \leq k$.
- **Assumption**: Jobs are independent and are requested from k parallel users, where $k \geq 2$.

Illustration of MUMPOSP. For simplicity and basic understanding of the readers, we illustrate an instance of *MUMPOSP* for scheduling of n jobs that are submitted by k users in Fig. 1. Here, $\{M_1, M_2, \ldots, M_m\}$ represent m identical machines and $\langle U_1, U_2, \ldots, U_{k-1}, U_k \rangle$ denote job sequences for k users, where each user has $\frac{n}{k}$ number of jobs. Jobs are submitted in batches online, where a batch is constructed after receiving exactly one job from each user (as long as a user has an unscheduled job). A batch consists of at least one job. Therefore, we have at least 1 batch, where $k = n$ and at most $n - k + 1$ batches, where any one of the users U_r has $n_r = n - k + 1$, and remaining users have exactly one job each. Each U_r seeks to obtain a minimum value for its makespan ($C_{\max}^r$) as the output, rather than the overall makespan ($C_{\max}$) of the system. Hence, it is indispensable for the scheduler to be fair while optimizing the $C_{\max}^r$, $\forall U_r$.

Representation of MUMPOSP. By following general framework $\alpha|\beta|\gamma$ of Graham et al. [4], we represent MUMPOSP as MUMPOSP(k, $P_m|C_{\max}^r$), where P_m denotes m identical machines and k represents number of users.

Perspectives of Fairness. Fairness has been considered and studied as a major performance criterion for scheduling algorithms in multi-user systems [5, 6] from two perspectives such as allocation of resources to the users and user's objective. Fairness of an algorithm with respect to resource allocation guarantees uniform allocation

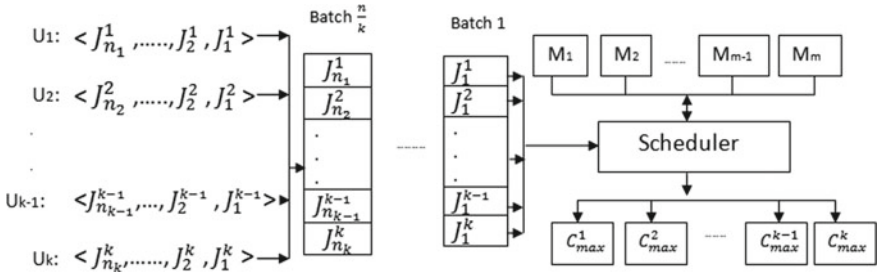

Fig. 1 Illustration of MUMPOSP for k users with equal number of jobs

of resources to the competing users [7]. The resources to be shared are application-dependent. For example, in client-server networking, the resources such as link bandwidth, network delay and specific time quantum can be shared [8, 9], whereas in case of *HPC* systems, the resources such as processors, memory and time slices can be shared [10, 11].

Algorithmic fairness based on user's objective is evaluated by the objective values achieved for respective users. An equality in the obtained objective values for a user ensures fairness of a scheduling algorithm. It is important for a fairness measure to define the equality for quantifying gap of an achieved objective value from the defined equality.

Related Work. Fairness as a quantitative performance measure based on resource allocation was studied by Jain et al. [7]. A set of properties for an ideal fairness measure was defined, and a fairness index $F(x)$ was proposed for resource allocation schemes. $F(x)$ is defined as follows: if any scheduling algorithm assigns resources to k competing users such that rth user gets an allocation of x_r. Then,

$$F(x) = \frac{(\sum_{r=1}^{k} x_r)^2}{\sum_{r=1}^{k} {x_r}^2}, \quad \text{where } x_r \geq 0.$$

The value of $F(x)$ is bounded between 0 and 1 to show percentage of fairness and discrimination of a resource allocation scheme for each user. Fairness based on sharing of resources such as processors, memory, system clock and system bus in multi-programmed multi-user system was well studied in [10–12]. Some recent works on fairness in scheduling online jobs on multi-user systems can be found in [13, 14]. To the best of our knowledge, fairness of online scheduling algorithms based on user's objective has not been exhaustively studied and explored the literature.

In [15–18], *stretch* matrix has been considered as a user's objective-based fairness measure for resource scheduling algorithms in multi-user systems. Here, *Stretch* (d_A^r) has been defined as a degradation factor in the objective value obtained by any algorithm A for each user U_r. Let us consider V_A^r be the objective value achieved by algorithm A and V_{OPT}^r be the optimum objective value for respective U_r. Then, stretch has been defined as follows:

$$d_A^r = \frac{V_A^r}{V_{\text{OPT}}^r}$$

The objective of any scheduling algorithm is to incur an equal stretch for each U_r to ensure fairness. Stretch matrix guarantees fairness based on equality in achieved objective values. However, it fails to depict the exact value of fairness per user as well as overall fairness of a scheduling algorithm. Stretch matrix does not capture the discrimination of a scheduling algorithm for the deprived users. Therefore, it is quintessential to define a formal fairness measure based on user's objective.

Our Contributions. We propose a novel model to evaluate fairness of online algorithms in the *Multi-user Multi-processor Online Scheduling Problem (MUMPOSP)*. We introduce and formally define quantitative fairness measures in our proposed model by considering optimization of makespan as user's objective. Furthermore, we define unfairness and absolute fairness of an online scheduling algorithm. We obtain lower bound results for the absolute fairness for a framework of m identical machines with equal length jobs. We show that our proposed model can be served as a framework for measuring algorithmic fairness by considering other optimality criteria such as sum of completion times, weighted sum of completion times and sum of flow times.

2 Our Proposed Fairness Model

We develop a new model, in which we define five quantitative measures to ensure algorithmic fairness. Instead of considering the resource allocation at the input level, our model considers the achieved value of user's makespan at the output level to determine the fairness of a scheduling algorithm. The model captures the issues of relative and global parameters for fairness by a *Fairness Index (FI)*. The issues of unfairness is captured by a *Discrimination Index (DI)*. The *FI* includes fairness parameters such as *Relative Fairness (RF)* and *Global Fairness (GF)*. Higher value of any fairness parameter indicates more fair algorithm. The *DI* includes unfairness measures such as *User Discrimination Index (UDI)*, *Global Discrimination Index (GDI)* and *Relative Discrimination Index (RDI)*. Lower value of any unfairness measure indicates higher degree of fairness of the algorithm. Before defining fairness and unfairness parameters, we illustrate our novel model and discuss the characteristics of a good fairness model as follows.

Illustration of Our Proposed Fairness Model. We illustrate our proposed fairness model as shown in Fig. 2. The model quantitatively defines the fairness of an online scheduling algorithm by taking into account the makespan ($C_{\max}^r$) of individual user in the *MUMPOSP* setup.

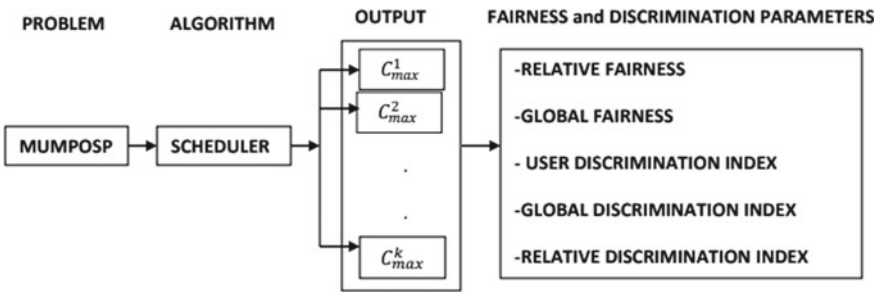

Fig. 2 A fairness model based on user's objective

2.1 Characteristics of a Good Fairness Model

A fairness model evaluates the performance of a scheduling algorithm based on the achieved makespan for each user. Recall that in [15–18], *Stretch* was considered as a user's objective-based fairness measure. For instance, if a scheduling algorithm A obtains makespans for three users as 5, 10, and 15, respectively, where their respective optimum makespans are 1, 5 and 10, then *stretch* defines the following degradation factors for respective users: $d_A^1 = 5$, $d_A^2 = 2$, and $d_A^3 = 1.5$.

Before formally defining fairness and unfairness parameters, we present the characteristics of a good fairness model as follows. A good fairness model must be:

- *Finitely Bounded*—The fairness of a scheduling algorithm is bounded within a finite interval, preferably between 0 and 1 for meaningful representation of fairness with respect to each user.
- *Consistent*—If any change in the scheduling policy results in different makespans for at least one user, then the change in the fairness parameters must be reflected for the concerned users as well as in the overall fairness of the policy.
- *Independent of Input Size*—It is applicable to any number of users with any number of jobs and machines.
- *Independent of Scale*. It must be able to measure fairness irrespective of units of measurement of processing time of the jobs such as seconds or milliseconds, microseconds or nanoseconds. The measuring unit must be uniform or inter convertible.

In addition to the above-mentioned properties, we also consider relative and overall fairness as an essential feature to develop our fairness parameters. We believe that the model must represent *relative equality* among achieved objective values for the users to show fairness of an algorithm for each user. For example, the users may not seek equal makespan as a gesture of fairness; however, they expect from an online scheduling algorithm to obtain an equal ratio between the *desired makespan (optimum value)* to the achieved makespan for all users. The value obtained by an algorithm for relative equality leads to *relative fairness* with respect to each user.

Also, the model must show *overall fairness* of an algorithm with respect to all users, which can lead to the comparison of the fairness of different scheduling policies.

2.2 Our Proposed Fairness and Unfairness Parameters

By considering the above-mentioned desirable properties, we now define formal measures of fairness and unfairness for MUMPOSP as follows.

Let A be an online scheduling algorithm. If algorithm A schedules jobs of k competing users on m identical processors such that rth user obtains a makespan of C_A^r, then we define the following fairness parameters.

Definition 1 The **Relative Fairness (RF)** obtained by algorithm A for any user U_r is defined as:

$$\mathrm{RF}(C_A^r) = \frac{C_{\mathrm{OPT}}^r}{C_A^r}, \quad \text{where } C_{\mathrm{OPT}}^r = \frac{\sum_{i=1}^{n_r} p_i^r}{m} \tag{1}$$

Corollary 1 *The **Relative Fairness Percentage (RFP)** for any user U_r obtained by algorithm A is defined as:*

$$RFP(C_A^r) = RF(C_A^r) \cdot 100 \tag{2}$$

Definition 2 The **Global Fairness (GF)** of algorithm A for k users is defined as:

$$\mathrm{GF}(C_A, k) = \frac{1}{k} \cdot \sum_{r=1}^{k} (\mathrm{RF}(C_A^r)) \tag{3}$$

Corollary 2 *The **Global Fairness Percentage (GFP)** of any algorithm A for k users is defined as:*

$$GFP(C_A, k) = GF(C_A, k) \cdot 100 \tag{4}$$

If algorithm A schedules jobs of k competing users such that rth user obtains a makespan of C_A^r, then we define **Fairness Index** for algorithm A represented by 2-tuple with two parameters such as RF and GF as follows

$$\mathrm{FI}(C_A, k) = \langle \{\mathrm{RF}(C_A^r) | 1 \leq r \leq k\}, \mathrm{GF}(C_A, k) \rangle \tag{5}$$

Example 1 Let us consider three departments {CSE, MAT, PHY} of a University as three users $\{U_1, U_2, U_3\}$, submitting jobs by MUMPOSP model to a centralized supercomputer (having 2 identical machines) in order to finish their respective projects at the earliest. Let us denote the job sequences of U_1, U_2, and U_3 as $U_1 = \langle J_1^1/1, J_2^1/2 \rangle$, $U_2 = \langle J_1^2/3, J_2^2/4 \rangle$ and $U_3 = \langle J_1^3/5, J_2^3/6 \rangle$, respectively. Suppose that the supercomputer runs an online scheduling algorithm *Alg* that schedules the jobs of U_1, U_2 and U_3 and obtains $C_{\mathrm{Alg}}^1 = 11$, $C_{\mathrm{Alg}}^2 = 9$ and $C_{\mathrm{Alg}}^3 = 10$, then we

have, $\mathrm{RF}(C^1_{\mathrm{Alg}}) = \frac{1.5}{11} = 0.13$ and $\mathrm{RFP}(C^1_{\mathrm{Alg}}) = 13\%$, $\mathrm{RF}(C^2_{\mathrm{Alg}}) = \frac{3.5}{9} = 0.38$ and $\mathrm{RFP}(C^2_{\mathrm{Alg}}) = 38\%$, $\mathrm{RF}(C^3_{\mathrm{Alg}}) = \frac{5.5}{10} = 0.55$ and $\mathrm{RFP}(C_{\mathrm{Alg}}) = 55\%$. Therefore, we have $\mathrm{GF}(C_A, 3) = 0.35$ and $\mathrm{GFP}(C_A, 3) = 35\%$.

Definition 3 The **Unfairness** of algorithm A for MUMPOSP with respect to each user U_r is defined by **User Discrimination Index** as:

$$\mathrm{UDI}^r_A = 1 - \mathrm{RF}(C^r_A) \tag{6}$$

Definition 4 The **Overall Unfairness** of algorithm A for k users is defined by **Global Discrimination Index** as:

$$\mathrm{GDI}(C^r_A, k) = 1 - \mathrm{GF}(C^r_A, k) \tag{7}$$

Definition 5 The **Realtive Discrimination Index (RDI)** of any algorithm A for MUMPOSP with respect to each user U_r is defined as:

$$\mathrm{RDI}^r_A = \begin{cases} \mathrm{GF}(C^r_A, k) - \mathrm{RF}(C^r_A), & \text{if } \mathrm{RF}(C^r_A) < \mathrm{GF}(C^r_A, k) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

If algorithm A schedules jobs of k competing users such that rth user obtains a makespan of C^r_A, then we define **Discrimination Index** for algorithm A as 3-tuple with three parameters such as UDI, GDI and RDI as follows.

$$\mathrm{DI}(C_A, k) = \langle\ \{\mathrm{UDI}^r_A \mid 1 \le r \le k\}, \mathrm{GDI}(C^r_A, k), \{\mathrm{RDI}^r_A | 1 \le r \le k\}\ \rangle \tag{9}$$

Example 2 Let us consider algorithm A results in relative fairness for U_1, U_2, U_3 and U_4 as 0.6, 0.6, 0.6 and 0.2 respectively. We now have $\mathrm{GF}(C^r_A, 4) = 0.5$. Therefore, $\mathrm{UDI}^1_A = 1 - 0.6 = 0.4$, $\mathrm{UDI}^2_A = 1 - 0.6 = 0.4$, $\mathrm{UDI}^3_A = 1 - 0.6 = 0.4$, $\mathrm{UDI}^4_A = 1 - 0.2 = 0.8$, $\mathrm{GDI}(C^r_A, 4) = 1 - 0.5 = 0.5$ and $\mathrm{RDI}^4_A = 0.5 - 0.2 = 0.3$.

3 Absolute Fairness and Lower Bound Results

We define absolute fairness as a quantitative measure and provide lower bound results of absolute fairness in generic *MUMPOSP* setting with equal length jobs. Let A be an online scheduling algorithm for the setup MUMPOSP $(k, P_m | C^r_{\max})$.

Definition 6 Algorithm A achieves **Absolute Fairness** if $\mathrm{RF}(C^r_A)$ is same $\forall U_r$, where $1 \le r \le k$.

Lemma 1 *If any algorithm A incurs* $RDI^r_A = 0$, $\forall U_r$, *then it achieves absolute fairness.*

Proof If $\mathrm{RDI}_A^r = 0, \forall U_r, 1 \leq r \leq k$, then by Eq. (8), we have

$$\mathrm{RF}(C_A^r) \geq \mathrm{GF}(C_A^r, k) \tag{10}$$

By Eqs. (3) and (10), we can infer that

$$\mathrm{RF}(C_A^r) = \mathrm{GF}(C_A^r, k), \forall U_r.$$

Therefore, Lemma 1 holds true. □

Definition 7 Any Algorithm A is b-**fair**, if it achieves $\mathrm{RF}(C_A^r) = b$ for all U_r, where $1 \leq r \leq k$ and $0 < b \leq 1$.

Theorem 1 *Any online algorithm A achieves absolute fairness in the setup MUMPOSP* $(k, P_2|C_{\max}^r)$ *such that* $\frac{C_{OPT}^r}{C_A^r} \geq \frac{1}{k}, \forall U_r$, *where* $k \geq 2$ *and* $1 \leq r \leq k$.

Proof Let us consider an instance of MUMPOSP $(k, P|C_{\max}^r)$, where $k = 2$. We analyze two cases based on n_r as follows.

Case 1: $n_1 \neq n_2$.
Case 1(a): If the first job pair (J_1^1, J_1^2) is scheduled on different machines. Let us consider the following instance $U_1 : \langle J_2^1/2, J_1^1/1\rangle, U_2 : \langle J_1^2/1\rangle$, where each job is specified by its processing time. Assigning $J_1^1/1$ and $J_1^2/1$ to machines M_1 and M_2, respectively, followed by the assignment of $J_2^1/2$ to either of the machines such that $C_A^1 = 3$ and $C_A^2 = 1$, where $C_{\mathrm{OPT}}^1 \geq 1.5$ and $C_{OPT}^2 \geq 0.5$. Therefore, we have $\frac{C_{\mathrm{OPT}}^1}{C_A^1} \geq \frac{1}{2}$ and $\frac{C_{\mathrm{OPT}}^2}{C_A^2} \geq \frac{1}{2}$.
Case 1(b): If the first job pair (J_1^1, J_1^2) is scheduled on the same machine. Let us consider the following instance $U_1 : \langle J_3^1/2, J_2^1/1, J_1^1/1\rangle, U_2 : \langle J_2^2/2, J_1^2/1\rangle$. If the first job pair $(J_1^1/1, J_1^2/1)$ is scheduled either on machine M_1 or on M_2, then by assigning the next pair of jobs (J_2^1, J_2^2) to the same or different machines, followed by the assignment of job $J_3^1/2$ such that $C_A^1 = 4$ and $C_A^2 = 3$, where $C_{\mathrm{OPT}}^1 \geq 2$ and $C_{\mathrm{OPT}}^2 \geq 1.5$. Therefore, we have $\frac{C_{\mathrm{OPT}}^1}{C_A^1} \geq \frac{1}{2}$ and $\frac{C_{\mathrm{OPT}}^2}{C_A^2} \geq \frac{1}{2}$.
Case 2: $n_1 = n_2$.
Case 2(a): If the first job pair (J_1^1, J_1^2) is scheduled on different machines. Let us consider the following instance $U_1 : \langle J_3^1/2, J_2^1/1, J_1^1/1\rangle, U_2 : \langle J_3^2/2, J_2^2/2, J_1^2/1\rangle$. Assigning jobs $J_1^1/1$ and $J_1^2/1$ to machines M_1 and M_2 respectively, followed by the assignment of the subsequent jobs as shown in Fig. 3a, such that $C_A^1 = 4$ and $C_A^2 = 5$, where $C_{\mathrm{OPT}}^1 \geq 2$ and $C_{\mathrm{OPT}}^2 \geq 2.5$. Therefore, we have $\frac{C_{\mathrm{OPT}}^1}{C_A^1} \geq \frac{1}{2}$ and $\frac{C_{\mathrm{OPT}}^2}{C_A^2} \geq \frac{1}{2}$.
Case 2(b): If the first job pair (J_1^1, J_1^2) is assigned to the same machine. We consider the same instance of Case 2(a). Assigning $J_1^1/1$ and J_1^2 on either machine M_1 or on M_2, followed by the assignment of the subsequent jobs as shown in Fig. 3b such that $C_A^1 = 4$ and $C_A^2 = 5$. Therefore, we have $\frac{C_{\mathrm{OPT}}^1}{C_A^1} \geq \frac{1}{2}$ and $\frac{C_{\mathrm{OPT}}^2}{C_A^2} \geq \frac{1}{2}$. □

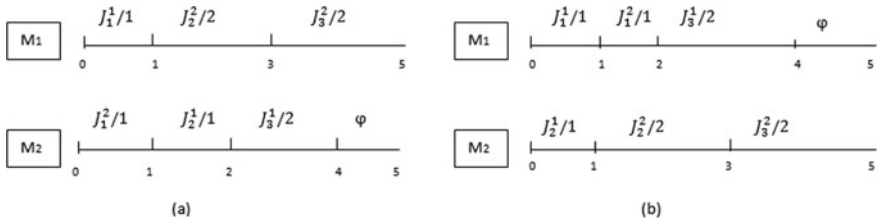

Fig. 3 Illustration of case 2

3.1 Results on Absolute Fairness in MUMPOSP with m Identical Machines for Equal Length Jobs

For ease of understanding, we analyze the lower bound of absolute fairness for any online algorithm in a generic *MUMPOSP* setting, where each user has equal number of jobs, and all jobs have equal processing time of x unit, where $x \geq 1$. The objective of each user is to obtain a minimum $C^r_{\max}$. We formally denote the problem as MUMPOSP $(k, P_m|p^r_i = x|C^r_{\max})$.

Lemma 2 *Let A be an online scheduling algorithm. In MUMPOSP* $(k, P_m|p^r_i = x|C^r_{\max})$ *with* $k = b \cdot m$, *algorithm A obtains* $C^r_A \leq b \cdot \sum_{i=1}^{n_r} p^r_i$, *for each* U_r, *respectively, where* $1 \leq r \leq k$, $m \geq 2$ *and* $b \geq 1$.

Proof We prove Lemma 2 by method of induction on number of jobs per user (n_r) as follows.

Induction Basis: Let us consider $k = m = 2, n_1 = n_2 = 1$ and $p^1_1 = p^2_1 = 1$.

Clearly, $C^r_A = 1 \leq b \cdot 1 \cdot 1$, where $r = 1, 2$ and $b \geq 1$.

Induction Hypothesis: Let us consider $k = b \cdot m, n_r = \frac{n}{k} = y$, where $y \geq 1, b \geq 1$ and $n = \sum_{r=1}^{k} n_r$.

We assume that

$$C^r_A \leq b \cdot \sum_{i=1}^{n_r} p^r_i \leq b \cdot x \cdot y \tag{11}$$

Inductive Step: For $n_r = y + 1$ with $p^r_i = x, \forall J^r_i$. We have to show that $C^r_A \leq (y + 1) \cdot b \cdot x$.

By Eq. (11), we have $C^r_A = y \cdot b \cdot x$ with $n_r = y$. When we add extra one job to each user, we have by *Induction Basis* $C^r_A = b \cdot x \cdot y + (b \cdot x) = (y + 1) \cdot b \cdot x$. Therefore, Lemma 2 holds true. □

Lemma 3 *Any algorithm A is* $\frac{1}{k}$*-fair for MUMPOSP* $(k, P_m|p^r_i = x|C^r_{\max})$ *with* $k = b \cdot m$, *where* $m \geq 2$ *and* $b \geq 1$.

Proof By Lemma 2, we have

$$C^r_A \leq b \cdot \sum_{i=1}^{n_r} p^r_i, \forall U_r \tag{12}$$

We have the fair optimum bound as

$$C^r_{\text{OPT}} \geq \frac{\sum_{i=1}^{n_r} p^r_i}{m}, \quad \forall U_r \tag{13}$$

By Eqs. (12) and (13), we have

$$\frac{C^r_{\text{OPT}}}{C^r_A} \geq \frac{1}{k}, \ \forall U_r. \tag{14}$$

Therefore, *Lemma* 3 holds true. □

Lemma 4 *In MUMPOSP* $(k, P_m|p^r_i = x|C^r_{\max})$ *with* $k > m$, *algorithm A obtains* $C^r_A \leq \lceil \frac{n}{m} \rceil \cdot x$, *for each* U_r *respectively, where* $k \neq m \cdot b$ *for* $b \geq 1$.

Proof The correctness of Lemma 4 is shown by method of induction on n_r as follows.

Induction Basis: Let us consider $m = 2, k = 3, n_r = 1$ and $p^r_i = 1$. Now, we have $n = n_r \cdot k = 3$.

Clearly, $C^r_A \leq 2 = \lceil \frac{n}{2} \rceil \cdot 1$.

Induction Hypothesis: Let us consider $n_r = \frac{n}{k} = y$, $p^r_i = x$ and $k > m$ with $k \neq m \cdot b$ for $b \geq 1$. We assume that $C^r_A \leq \lceil \frac{n}{m} \rceil \cdot x, \forall U_r$.

Inductive Step: We show that $C^r_A \leq \lceil \frac{n+k}{m} \rceil \cdot x$ for $n_r = y + 1, \forall U_r$.

By our Induction Basis, for one extra job of each user U_r, where $1 \leq r \leq k$, algorithm A incurs an additional time of $\lceil \frac{k}{m} \rceil \cdot x$ for each U_r.

Therefore, $C^r_A \leq \lceil \frac{n}{m} \rceil \cdot x + \lceil \frac{k}{m} \rceil \cdot x \leq \lceil \frac{n+k}{m} \rceil \cdot x$

Thus, Lemma 4 holds true. □

Theorem 2 *Any Algorithm A is* $\frac{1}{k}$*-fair for MUMPOSP* $(k, P_m|p^r_i = x|C^r_{\max})$, *where* $k \geq m$ *and* $m \geq 2$.

Proof Theorem 2 holds true by Lemma 3 for $k = m \cdot b$, where $b \geq 1$.

By Lemma 4, we have

$$C^r_A \leq \lceil \frac{n}{m} \rceil \cdot x \tag{15}$$

By Eq. (13), we have $C^r_{\text{OPT}} \geq \frac{\frac{n}{k} \cdot x}{m}$.

Implies,

$$C^r_{\text{OPT}} \geq \frac{n \cdot x}{k \cdot m} \tag{16}$$

By Eqs. (14) and (15), we have

$$\frac{C^r_{\text{OPT}}}{C^r_A} \geq \frac{\frac{n \cdot x}{k \cdot m}}{\frac{n \cdot x}{m}}$$
$$\geq \frac{n \cdot x \cdot m}{n \cdot k \cdot m \cdot x} \geq \frac{1}{k}. \qquad \square$$

4 Fairness Measure Using Flow Time and Completion Time as User's Objective

We show that our proposed Fairness Index can be served as a framework for measuring fairness of any algorithm based on well-known user's objectives such as *sum of completion times* (S^r), weighted sum of completion times (W^r) and sum of flow times (SF^r). Selection of an user's objective is application-dependent. For instance, users of interactive systems require optimized value for respective flow time f^r, where f_i^r of any J_i^r is the difference between its completion time c_i^r and arrival time t_i^r. We now define relative fairness measures based on the above-mentioned user's objectives, respectively, by our proposed *FI*.

- **Sum of Completion Times** (S^r): Here, the objective for each U_r is to obtain a minimum $S^r = \sum_{i=1}^{n_r} c_i^r$. The relative fairness for any U_r, obtained by any algorithm A based on S^r is defined as

$$R_A(S_A^r) = \frac{S_{\mathrm{OPT}}^r}{S_A^r}, \quad \text{where } S_{\mathrm{OPT}}^r \text{ is the optimum value for } S^r.$$

- **Weighted Sum of Completion Times** (W^r): Here, the c_i^r is associated with certain positive weight w_i^r. The objective for each U_r is to obtain a minimum $W^r = \sum_{i=1}^{n_r} w_i^r \cdot c_i^r$. The relative fairness for any U_r obtained by algorithm A based on W^r is defined as

$$R_A(W_A^r) = \frac{W_{\mathrm{OPT}}^r}{W_A^r} \quad \text{where, } W_{\mathrm{OPT}}^r \text{ is the optimum value for } W^r.$$

- **Sum of Flow Times** (SF^r): Here, each U_r wants a minimum value for respective $\mathrm{SF}^r = \sum_{i=1}^{n_r} f_i^r$, where f_i^{*r} is the desired value of f_i^r and $\mathrm{SF}_{\mathrm{OPT}}^r = \sum_{i=1}^{n_r} f_i^{*r}$. The relative fairness for any U_r obtained by algorithm A based on SF^r is defined as

$$R_A(\mathrm{SF}_A^r) = \frac{\mathrm{SF}_{\mathrm{OPT}}^r}{\mathrm{SF}_A^r}.$$

5 Concluding Remarks and Scope of Future Work

In this work, we make an attempt to address the non-trivial research challenge of defining a new fairness model with quantitative measures of algorithmic fairness for *Multi-user Multi-processor Online Scheduling Problem (MUMPOSP)* based on user's objective. We formally presented the *MUMPOSP* setting with an illustration followed by a discussion on perspectives of fairness in *MUMPOSP*. We have proposed a new fairness model and have defined five quantitative measures to ensure algorithmic fairness by considering minimization of makespan as the user objective.

Lower bound results on absolute fairness of an online scheduling algorithm have been shown in *MUMPOSP* setup with equal length jobs. We have shown how our proposed fairness measure can be served as a framework for measuring algorithmic fairness based on well-known user's objectives such as sum of completion times, weighted sum of completion times and sum of flow times.

Scope of Future Work. We assumed a ideal theoretical bound for C^r_{OPT}. It is still open to explore a realistic bound for C^r_{OPT}. A non-trivial challenge is to compare the fairness of any two online scheduling algorithms A and B, when global fairness of algorithms A and B are same, whereas relative fairness of A is more than that of B for some users or vice-versa. In this scenario, it is interesting to make a trade-off by considering the number of users and individual relative fairness for each user to compare the fairness of two different algorithms.

References

1. Emmott S, Rison S (2020) Towards 2020 science. Tech. Rep., Microsoft Research Cambridge, Working Group Research
2. Jackson D, Snell Q, Clement M (2001) Core algorithms of the Maui scheduler. In: Feitelson DG, Rudolph I (eds) 7th international workshop JSSPP. LNCS, vol 2221. Springer, Heidelberg
3. Anderson DP (2004) BOINC: a system for public-resource computing and storage. In: 5th IEEE/ACM international workshop on grid computing, pp 4–10
4. Graham RL, Lawer EL, Lenstra JK, Rinnooy kan AH (1979) Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann Discrete Math 5:287–326
5. Jaffe JM (1980) A decentralized optimal multiple user flow control algorithm. In: Proceedings of the international conference computer communications, Atlanta, GA
6. Jaffe JM (1981) Bottleneck flow control. IEEE Trans Commun COM-29(7):954–962
7. Jain RK, Chiu DMW, Hawe WR (1984) A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA, TR-301
8. Bharath-Kumar K, Jaffe JM (1981) A new approach to performance-oriented flow control. IEEE Trans Commun COM-29(4):427–435
9. Sauve JP, Wong JW, Field JA (1980) On fairness in packet switching networks. In: Proceedings of the 21st IEEE Computer Society international conference, COMPCon 80, Washington, DC, pp 466–470
10. Kay J, Lauder P (1988) A fair share scheduler. Commun ACM 31(1):44–55
11. Feitelson DG (1997) Job scheduling in multi-programmed parallel systems (extended version). IBM Research Report, RC19790(87657), Second Revision
12. Vandierendonck H, Seznec A (2011) Fairness metrics for multi-threaded processors. IEEE Comput Archit Lett 10(1):4–7
13. Sun H, Hsu WJ, Cao Y (2014) Competitive online adaptive scheduling for sets of parallel jobs with fairness and efficiency. J Parallel Distrib Comput 74(3):2180–2192
14. Bian S, Huang X, Shao Z (2019) Online task scheduling for fog computing with multi-resource fairness. In: Proceedings of the 90th IEEE vehicular technology conference, Honolulu, HI, USA, pp 1–5
15. Bender MA, Muthukrishnan S, Rajaraman R (2002) Improved algorithms for stretch scheduling. In: Proceedings of the 13th annual ACM-SIAM symposium on discrete algorithms (SODA), pp 762–771

16. Legrand A, Su A, Vivien F (2006) Minimizing the stretch when scheduling flows of biological requests. In: Symposium on parallelism in algorithms and architectures (SPAA)
17. Saule E, Trystram D (2009) Multi users scheduling in parallel systems. In: Proceedings of IEEE international parallel and distributed processing symposium, Washington, DC, USA, pp 1–9
18. Pinheiro VG (2014) The management of multiple submissions in parallel systems: the fair scheduling approach. PhD Thesis, Institute of Mathematics and Statistics, University of Sao Paulo, Brazil