ON THE MINIMIZATION OF WORKLOAD BALANCING CRITERIA ON IDENTICAL PARALLEL MACHINES

Anthony Cossari, Dipartimento di Economia e Statistica, Universita della Calabria, 87036 Arcavacata di Rende (CS), Italy. a.cossari@unical.it (39) 0984-492431

Johnny C. Ho, Department of Management and Marketing, Turner College of Business and Computer Science, Columbus State University, Columbus, GA 31907, USA. ho_johnny@columbusstate.edu (706) 507-8087

Giuseppe Paletta, Dipartimento di Economia e Statistica, Universita della Calabria, 87036 Arcavacata di Rende (CS), Italy. g.paletta@unical.it (39) 0984-492428

Alex J. Ruiz-Torres, Departamento de Gerencia, Facultad de Administración de Empresas, Universidad de Puerto Rico – Rio Piedras, San Juan, PR 00931, USA. alex.ruiztorres@uprrp.edu (787) 764-0000

ABSTRACT

This paper studies the problem of scheduling jobs on identical parallel processors to minimize workload balancing criteria. While workload balancing is certainly an important criterion given the need of production systems to use all of their resources efficiently, there is no established measure of performance in the scheduling literature that characterizes total workload balance. In this paper, we propose the normalized standard deviation, the normalized mean deviation, and the normalized mean difference as criteria that measure the balance of the workloads among the machines associated to a schedule. A local search heuristic, which performs multiple exchanges of jobs among machines, is presented.

Keywords: Parallel machines scheduling, workload balancing criteria, measures of dispersion, heuristics.

1. INTRODUCTION

In the classical multiprocessor scheduling problem, a set \( J = \{1, ..., j, ..., n\} \) of \( n \) simultaneously available independent jobs, with a non-negative processing times \( p_j > 0, j = 1, ..., n \), are scheduled on a set \( M = \{1, ..., i, ..., m\} \) of \( m \) identical parallel machines. Each machine can process at most one job at a time, and each job must be processed without interruption by exactly one of the machines. It is supposed that \( n > m \geq 2 \) to avoid trivialities. A schedule (solution) is represented by an \( m \)-partition \( S = \{S_1, ..., S_i, ..., S_m\} \) of the set \( J \), where each \( S_i \) represents the subset of jobs assigned to the machine \( i, i = 1, ..., m \). For each schedule \( S \), the work-loads of the machines are represented by the \( m \)-set \( C(S) = \{C(S_1), ..., C(S_i), ..., C(S_m)\} \), where \( C(S_i) = \sum_{j \in S_i} p_j \) is the workload of machine \( i, i = 1, ..., m \).

Workload balancing is an important practical criterion given the need of production systems to
efficiently use all of their resources. When the work loads are not balanced, resources will become idle waiting for the next set of jobs, for example, a common setup is required for all the parallel resources. It is therefore highly desirable in many production environments with parallel processing to finish the last assigned job for each resource at the same time. However, there is no “established” measure of performance in the scheduling literature that characterizes workload balance. In this paper, the normalized standard deviation, the normalized mean deviation, and the normalized mean difference are proposed as criteria that measure the balance of the workloads among the machines associated to a schedule $S$.

The paper describes the characteristics of each of the proposed measures of workload balance and the need for normalization when multiple instances will be performed as part of an experiment. It also presents an algorithm aimed at workload balancing based on job exchanges. Experiments were performed to evaluate the performance of the algorithm and to determine differences among the criteria. The experiments demonstrated consistency across the three criteria despite the possibility of difference in schedule evaluation.

The remainder of the paper is organized as follows. In Section 2, we describe the three measures of performance for workload balancing rooted in dispersion measures. Section 3 describes an algorithm to generate schedules with minimal unbalance. Finally, concluding remarks and directions for future research are provided in Section 4.

### 2. PERFORMANCE CRITERIA AND STATISTICAL MEASURES OF DISPERSION

When scheduling jobs on identical parallel machines, an important practical objective is to find the optimal balance of the workloads among the machines, that is to allocate jobs in the machines in such a way that the resulting workloads are as close to each other as possible. Such workload balancing problem is nothing but the basic problem of number partitioning, where a given set of integers are assigned to a collection of subsets so that the sums of the numbers in each subset are as nearly equal as possible. Obviously, an appropriate performance criterion capable of measuring the extent to which balance is obtained needs to be selected. A natural way to approach this desideratum is to look at a measure of the workload deviations and to minimize it. Currently, a number of ad hoc criteria are available in both multiprocessor scheduling and number partitioning contexts. These criteria have been recently investigated by Cossari et al. [4], who demonstrated that they all are essentially well-known statistical measures of data dispersion, and also that two of them are closely related to each other. We now briefly present these existing criteria, then in Section 2.2 the choice of a performance criterion will be put into the broader perspective of choosing an appropriate dispersion measure from those available in statistical methodology.

#### 2.1 Existing Ad Hoc Criteria

Ho et al. [9] addressed the workload balancing problem in multiprocessor scheduling and proposed to measure balance via the \(NSSWD\) criterion, which is defined for schedule $S$ as

\[
W(S) = \frac{1}{\mu} \sum_{i=1}^{m} (C(S_i) - \mu)^2
\]
where $\mu = \left( \frac{1}{m} \sum_{i=1}^{m} C(S_i) \right) = \left( \frac{1}{m} \sum_{i=1}^{m} p_i \right)$ is the average of the $C(S_i)$'s, that is the mean completion time. This criterion stems from the well-established practice of measuring data dispersion in terms of the squared deviations from the mean, and has the additional feature of being suited for the relative assessment of multiple problem instances. Later Cossari et al. (2011) showed that $W(S) = \sqrt{m \sigma^2 / \mu}$, where

$$\sigma = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (C(S_i) - \mu)^2}$$

is the popular standard deviation and $\sigma / \mu$ is the coefficient of variation, which is typically adopted in statistical practice to compare dispersions in different groups of data. Thus, except for the constant $\sqrt{m}$, the NSSWD criterion is essentially equivalent to the coefficient of variation, implying that both these indices may be equally used as performance measures. Indeed, a little drawback of $W(S)$ is that it inflates $\sigma / \mu$ in an increasing way as $m$ increases. When a given problem instance is to be considered in practical applications, both $m$ and $\mu$ have fixed values and therefore the standard deviation $\sigma$ may just be used being equivalent to NSSWD as well as the coefficient of variation.

In the context of number partitioning, it is customary to obtain partitioning by minimizing the maximum pairwise difference between subset sums $C(S_i)$, namely:

$$\Delta(S) = \max_{i=1,\ldots,m} \{C(S_i)\} - \min_{i=1,\ldots,m} \{C(S_i)\}.$$

This objective function is just the range, which is a well-known, though poor measure of data variability in that it gives the length of the smallest interval that comprises all the data. Going beyond the usual approach for number partitioning, Alidaee et al. [1] proposed the following criterion:

$$x_0 = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} [C(S_i) - C(S_j)]^2$$

which allows to recast the problem as an unconstrained quadratic binary program that can be solved by efficient metaheuristic methods. Cossari et al. [4] showed that $x_0 = m^2 \sigma^2$, where $\sigma^2$ (square of the standard deviation) is the very popular variance, and hence, apart from a factor $m^2$, $x_0$ is essentially equivalent to the variance and to the standard deviation itself. When solving a single instance $m$ is fixed, thus the use of $x_0$ is unnecessary and $\sigma$ may again be equally employed. If multiple problem instances are to be examined for purposes of comparison, the use of $x_0$ should be avoided since it is affected by different values of $\mu$ so its results are not comparable to each other. Although the criteria $\Delta(S)$ and $x_0$ have been devised for number partitioning, it should be emphasized that they both might be equally used in the workload balancing problem for multiprocessor scheduling.

Furthermore, a remarkable, perhaps unexpected mathematical relationship between NSSWD and $x_0$ arises, which gives further insight into these criteria (see Cossari et al. [4] for details). In particular, for a given problem instance, $W(S)$ and $x_0$ turn out to be proportional to each other, thus they may be used interchangeably for both multiprocessor scheduling and number partitioning, together with their associated algorithms. In this situation, indeed, the standard
deviation $\sigma$ is an equivalent, more simple criterion, which therefore may definitely be used for both problems. Instead, when various instances need to be investigated, for example in comparative studies, the NSSWD criterion, or the equivalent coefficient of variation $\frac{\sigma}{\mu}$, are to be preferred to $x_0$ as they are unaffected by different values of $\mu$.

In the special case of $m = 2$ machines (subsets), all the aforementioned criteria reduce to some function of $|C(S_1) - C(S_2)|$ (Cossari et al. [4]), that is, the absolute difference between completion times (subset sums) in the two machines (subsets). Obviously, minimizing such a function means minimizing the makespan, thus the workload balancing problem is just a minimum makespan problem.

2.2 Criteria Derived from Dispersion Measures

As evidenced by the preceding subsection, any performance criterion for workload balancing is nothing but a statistical measure of dispersion. Moving away from the practice of generating an ad hoc criterion, the task of choosing an appropriate objective function is now viewed from a global perspective on dispersion measures, arguing that, in principle, any of such measures is eligible for use as a performance criterion. In what follows we give a systematic account of dispersion measures, focusing on the most suitable ones for our purposes (a comprehensive treatment may be found, e.g., in Stuart and Ord [11, pp.52–68]). Clearly, they are treated merely as descriptive indices of variability, thus not considering their inferential properties which are not relevant to our framework.

Dispersion measures may be grouped into three basic categories:
(a) Measures based on the difference between certain representative values of the dataset.
(b) Measures based on the deviations of every datum from some central value.
(c) Measures based on the differences of all the data among themselves.

The range of a dataset is the difference between its greatest and least values, thus belongs to category (a). As remarked before, such measure is the most typical objective function in the context of number partitioning, perhaps because of its simplicity. It may be a not very accurate measure of dispersion though, as it simply gives the distance between the extreme values thus ignoring how the bulk of the dataset varies inside the range. It seems that its use in number partitioning or multiprocessor scheduling problems has a poor justification, and hence some more precise measure of variability should be employed instead. Some other measures fall into group (a), such as the interquartile range and the interdecile range (see Stuart and Ord [11, p.52] for more details). Although such measures can be slightly more informative than the range, they still give a rough assessment of dispersion thus seem to be not useful enough in our framework.

When considering measures of category (b), first some central value need to be specified, for example the mean or the median of the dataset. If the mean is used, then the variance naturally arises as the average of the squares of the deviations from the mean. Certainly, the variance is the dispersion measure par excellence in statistical inference, largely due to its straightforward mathematical tractability in the theory of sampling. By extension, it is usually employed for descriptive analysis as well, but a number of alternatives may be equally used in this case as we will show in the following. A drawback of the variance is that its value is expressed in the
squared unit of the data, making its interpretation hard. This is resolved by simply taking the positive square root of the variance thus obtaining the aforementioned standard deviation $\sigma$. As shown earlier, recent proposals of performance criteria for multiprocessor scheduling and number partitioning, namely NSSWD and $x_0$, prove to be some function of $\sigma$ (and the variance), suggesting that their use is unnecessary for single instance problems and the equivalent standard deviation may be used instead.

A different measure falling into group (b) arises if we consider taking the absolute values of the deviations from the mean rather the squares. This leads to:

$$\delta_1 = \frac{1}{m} \sum_{i=1,\ldots,m} |C(S_i) - \mu|$$

which is known as the mean deviation (from the mean). Clearly, this is a very intuitive measure of dispersion around the mean analogous to the variance (and the standard deviation), which may naturally be used as a descriptive index of variability. It has generally little use in inferential statistics because of the difficulty in handling the absolute value in the sampling theory. We, therefore, recommend its possible use as a performance criterion in our framework in addition to the standard deviation. It is worth noting that, in general, the mean deviation is not greater than the standard deviation, and some sharper inequalities may also occur in some cases (Stuart and Ord [11, p.56]).

Sometimes, the central value from which deviations are compiled is assumed to be the median, that is, the middle value in the ordered sequence of data. In this case, one measure in widespread use is the mean deviation from the median, which is the same as the mean deviation defined above, except that the median is used in place of the mean. Such measure has some attractive features which usually make it preferable to the mean deviation and even to the standard deviation, especially when outliers among the data are present. However, the median is a poor central value for our typical small set of $C(S_i)$'s (completion times or subset sums), thus the mean deviation from the median seems not appropriate for use in number partitioning and multiprocessor scheduling and will not be discussed further.

The main representative of category (c) is the mean difference (without repetition) defined in our context by:

$$\delta_2 = \frac{1}{m(m - 1)} \sum_{i=1,\ldots,m} \sum_{j=1,\ldots,m} |C(S_i) - C(S_j)|$$

If compared to indices belonging to group (b), such a measure stems from a completely different viewpoint. In fact, while quantities such as the standard deviation or the mean deviation are intended to measure the workload (subset sum) balance as closeness of the $C(S_i)$'s to the mean value, the mean difference does measure balance as closeness of the $C(S_i)$'s to each other. In particular, the mean difference provided above represents the average of the differences of all the possible pairs of $C(S_i)$ values regardless of their sign, where the divisor $m(m - 1)$ is the number of pairs given that each $C(S_i)$ is not taken with itself thus explaining why the resulting index is qualified as the mean difference without repetition. A variant is to consider the mean difference with repetition where each $C(S_i)$ is taken with itself also, but the sum of differences clearly remains unchanged and the only distinction lies in the divisor which is now $m^2$. Obviously, the
two measures are related to each other, thus the former will be just used here and referred to simply as the mean difference. This index was popularized in the statistical community by Gini [5], even if it was used since the 1870s in some astronomical studies (Stuart and Ord [11, p.58]). Like the mean deviation, it suffers from the same difficulty in the mathematical derivations in sampling theory, but it is a truly attractive descriptive measure for our purposes, mainly because it focuses on dispersion of the $C(S_i)$ among themselves which is the natural object in number partitioning or scheduling problems. Several expressions have been devised to simplify the calculation of the numerator of the mean difference; some of them are reported in Stuart and Ord [11, p.62]. Besides, it may be shown that the mean difference cannot exceed $\sqrt{2}$ times the standard deviation. One might argue that a further measure may be obtained if the pairwise differences are squared rather than taken absolutely (this was done by Alidaee et al [1] with their criterion $x_0$). Unfortunately, such a measure is nothing but twice the variance, as already noted by Cossari et al. [4], thus it carries no theoretical value.

To summarize, apart from the range whose widespread use in number partitioning is highly questionable, three dispersion measures are regarded as fully adequate to quantify the balance of the $C(S_i)$ among the machines (subsets) and thus recommended as performance criteria. These are the standard deviation (or the variance), the mean deviation and the mean difference, all of which are expressed in the same unit of the data with the exception of the variance. Their use, however, should be restricted to problems where a single given instance is to be resolved. There are situations, in fact, where multiple instances need to be studied, for example when a simulation experiment is produced in order to compare different algorithms. In such cases, each of the suggested criteria has values for the various instances which are not fully comparable, being affected by the $\mu$ value arising in the associated instance (see also discussion in Cossari et al [4]).

There are two possible solutions to this comparability problem. One approach is to consider dividing by $\mu$ to obtain the ratios $\frac{\sigma}{\mu}$, $\frac{\delta_1}{\mu}$, and $\frac{\delta_2}{\mu}$, which are pure numbers clearly unaffected by the mean and thus appropriate for purposes of comparison. The first of such relative measures, the coefficient of variation, is the best-known one and is widely used in statistical practice for comparing variability. We recall that it coincides essentially with the $\text{NSSWD}$ criterion introduced by Ho et al. [9]. Another approach to obtain normalized measures ranging between 0 and 1 is by dividing the standard deviation, the mean deviation, and the mean difference by their maximum value, respectively. Such a maximum is the value attained in the assumption of maximum variability, which occurs for a hypothetical schedule where a single machine (subset) has the entire workload (sum of numbers) while the other $m-1$ machines (subsets) have no workload (sum of numbers) at all. By direct calculation it turns out that the following maxima arise:

$$\max \sigma = \mu \sqrt{m - 1}; \quad \max \delta_1 = 2 \mu \frac{m-1}{m}; \quad \max \delta_2 = 2 \mu,$$

which generate three normalized criteria as follows:

$$\sigma_{\text{norm}} = \frac{\sigma}{\mu \sqrt{m - 1}}; \quad \delta_{1\text{norm}} = \frac{m \delta_1}{2(m-1)\mu}; \quad \delta_{2\text{norm}} = \frac{\delta_2}{2\mu}.$$ 

We observe from these expressions that they may also be interpreted as the normalized variants of the relative measures defined earlier. Obviously, these criteria increase towards 1 as
dispersion of the $C(S_i)$’s increases and, like all the preceding measures, have the ideal value of 0 in case of a perfect balance with all of the $C(S_i)$’s equal to each other and thus equal to the mean $\mu$. Having obtained the schedule (partition) that minimizes $\sigma^{\text{norm}}, \delta_1^{\text{norm}}$ or $\delta_2^{\text{norm}}$, the normalized nature of these measures serves the purpose of suggesting how close is the given optimal schedule (partition) to the ideal schedule (partition) of perfect balance, thus assessing the extent to which balance was obtained. Therefore, these normalized performance criteria are especially recommended for multiple instance problems where comparisons need to be made, but also for single instance problems, and will be used later in our simulation experiment. Among them, the best-known in statistical practice is $\delta_2^{\text{norm}}$, which does coincide with the famous Gini’s coefficient of concentration, usually denoted by $R$, arising from a different perspective as a measure of inequality in studies on, e.g., income or wealth (e.g., see Stuart and Ord [11, pp.60–64]). In principle, other indices of inequality might be potentially useful in our context, but this possibility will not be explored further in this paper and is possibly deferred to subsequent work.

Finally, we mention the case of $m = 2$ machines (subsets). Once again, every new criterion proposed in this subsection readily proves to be some function of $|C(S_1) - C(S_2)|$, thus confirming that the problems of number partitioning and workload balancing reduce to the problem of minimum makespan whichever criterion is employed to measure balance.

### 3. ALGORITHM

In the following, we describe a local search algorithm named Work-load Balancing Algorithm ($WBA(w)$) for balancing the workloads of the machines with respect to the performance measure $w$. It may be applied with any of the related criteria discussed in Section 2, namely the normalized standard deviation, the normalized mean deviation, and the normalized mean difference, by setting $w = \sigma^{\text{norm}}, w = \delta_1^{\text{norm}}$ and $w = \delta_2^{\text{norm}}$, respectively.

$WBA(w)$ uses three local search procedures that are performed in the sequence. The first procedure, referred to as Job Interchange Procedure ($JIP(w)$), interchanges one or two jobs performed by a same machine with one or two jobs performed by another machine, if an advantage has been identified, i.e., $w$ decreases for effect of the interchange.

The second procedure is a slightly modified version of $JIP(w)$, referred to as $JIP1(w)$, where machines are sorted in non-increasing order of their workloads. Moreover, if $i$ and $j$ are the machines interested by the current exchange, the jobs in each of these two machines are sorted in non-increasing order of their processing times.

The third procedure, referred to as $JIP2(w)$, is a further modification of $JIP1(w)$, in which the jobs of machine $i$ are sorted in non-increasing order of their process times, while those of machine $j$ are sorted in non-decreasing order of their process times.

Our heuristic returns the best solution among those obtained by using all these procedures in sequence. Obviously, the optimum may be found before the entire sequence is completed. This certainly happens when $w = 0$. With the aim of leading to the better exploration of the solution space, the current solution is restored to the initial feasible solution, before second and third procedures are run. The following is a general schema of the proposed algorithm.
**WBA\(w\) Algorithm**

Step 1. Consider an initial feasible solution \(\hat{S}\), and compute \(w\). Set \(S_{\text{best}} = \hat{S}\) and \(w_{\text{best}} = w\). If \(w = 0\) (the current solution is optimal) then go to Step 5.

Step 2. Choose \(S = \hat{S}\) as starting point and perform \(\text{JIP}(w)\). If \(w_{\text{best}} > w\) then set \(S_{\text{best}} = S\) and \(w_{\text{best}} = w\). If \(w = 0\) then go to Step 5.

Step 3. Choose \(S = \hat{S}\) as starting point. Perform \(\text{JIP1}(w)\). If \(w_{\text{best}} > w\) then set \(S_{\text{best}} = S\) and \(w_{\text{best}} = w\). If \(w = 0\) then go to Step 5.

Step 4. Choose \(S = \hat{S}\) as starting point. Perform \(\text{JIP2}(w)\). If \(w_{\text{best}} > w\) then set \(S_{\text{best}} = S\) and \(w_{\text{best}} = w\). If \(w = 0\) then go to Step 5.

Step 5. Return \(S_{\text{best}}\) and \(w_{\text{best}}\).

In local search algorithms, an important decision is the selection of the initial solution, i.e., the initial assignment of the jobs to the machines. In the preliminary experimentation the algorithm ran starting from different solutions, obtained via LPT (Graham [6,7]) and MULTIFIT (Coffman et al. [3]) procedures for P||\(C_{\text{max}}\) (Graham et al. [8]). In general, LPT showed to be the best choice, and therefore it has been chosen for generating the starting solution.

### 3.1 Job Interchange Procedure (\(\text{JIP}(w)\))

Given a current feasible solution \(S\) and a minimal \(w\) scheduling problem, the procedure \(\text{JIP}(w)\), for each couple of machine, iteratively interchanges:

- two jobs, \(j \in S_i\) and \(k \in S_i\), from the machine \(i\) with two jobs, \(u \in S_l\) and \(v \in S_l\), from the machine \(l\), if the variation, say \(w_{u,v}^{j,k}\), of the criterion \(w\) identifies an advantage.
- the job, \(j \in S_i\), from the machine \(i\) with two jobs, \(u \in S_l\) and \(v \in S_l\), from the machine \(l\), if the variation, say \(w_{u,v}^j\), of the criterion \(w\) identifies an advantage.
- two jobs, \(j \in S_i\) and \(k \in S_i\), from the machine \(i\) with the job, \(u \in S_l\), from the machine \(l\), if the variation, say \(w_{u}^{j,k}\), of the criterion \(w\) identifies an advantage.
- the job, \(j \in S_i\), from the machine \(i\) with the job, \(u \in S_l\), from the machine \(l\), if the variation, say \(w_u^j\), of the criterion \(w\) identifies an advantage.

Obviously, the exchange is advantageous if such variations are negative thus resulting in a reduction in the \(w\) criterion. If the interchange has been carried out then it is necessary to update the current feasible solution.

The variations \(w_{u,v}^{j,k}\), \(w_{u,v}^j\), \(w_u^{j,k}\) and \(w_u^j\) are determined by setting \(w = \sigma^{\text{norm}}, w = \delta_1^{\text{norm}}\) and \(w = \delta_2^{\text{norm}}\). Formally, the procedure can be described as follows.

**\(\text{JIP}(w)\) Procedure**

Step 0. Consider the current feasible solution \(S = \{S_1, \ldots, S_m\}\) and the corresponding \(C(S) = C(S_1), \ldots, C(S_i), \ldots, C(S_m)\). Set \(i = 1\) and \(l = m\).

Step 1. Consider \(S_1\) and \(u \in S_i\)

\(v \in S_i\) and \(v \neq u\)

\(k \in S_i\) and \(k \neq j\)
If \((w_{i,j} < 0)\) then update the current feasible solution by setting \(S_i = S_i \setminus \{j,k\} \cup \{u,v\}\), \(S_i = S_i \setminus \{u,v\} \cup \{j,k\}\), \(C(S_i) = C(S_i) - p_j - p_k + p_u + p_v\), \(C(S_i) = C(S_i) - p_j - p_v + p_i + p_k\). If \(w = 0\) then go to Step 3 else set \(i = 1\) and \(l = m\), and go to Step 1;
End \(k\).

If \((w_{i,j} < 0)\) then update the current feasible solution by setting \(S_i = S_i \setminus \{j\} \cup \{u,v\}\), \(S_i = S_i \setminus \{u,v\} \cup \{j\}\), \(C(S_i) = C(S_i) - p_j + p_u + p_v\), \(C(S_i) = C(S_i) - p_u - p_v\). If \(w = 0\) then go to Step 3 else set \(i = 1\) and \(l = m\), and go to Step 1;
End \(v\).

\(k \in S_i\) and \(k \neq j\)
If \((w_{i,j} < 0)\) then update the current feasible solution by setting \(S_i = S_i \setminus \{j,k\} \cup \{u\}\), \(S_i = S_i \setminus \{u\} \cup \{j,k\}\), \(C(S_i) = C(S_i) - p_j - p_k + p_u + p_v\), \(C(S_i) = C(S_i) - p_u + p_j + p_k\). If \(w = 0\) then go to Step 3 else set \(i = 1\) and \(l = m\), and go to Step 1;
End \(k\).

If \((w_{i,j} < 0)\) then update the current feasible solution by setting \(S_i = S_i \setminus \{j\} \cup \{u\}\), \(S_i = S_i \setminus \{u\} \cup \{j\}\), \(C(S_i) = C(S_i) - p_j + p_u\), \(C(S_i) = C(S_i) - p_u + p_j\). If \(w = 0\) then go to Step 3 else set \(i = 1\) and \(l = m\), and go to Step 1;
End \(u\) and \(j\).

Step 2. If \(l > i\) set \(l = l - 1\) and go to Step 1, otherwise if \(i < m - 1\) set \(i = i + 1\) and \(l = m\) and go to Step 1, otherwise go to Step 3.

Step 3. Return \(S\), \(C(S)\) and \(w\).

With regard to efficiency, we note that the procedure considers the interchange of one or two jobs performed by a same machine with one or two jobs performed by a different machine. According to Hubscher and Glover [10] and Anderson et al. [2], the maximum possible neighborhood size occurs when the jobs are evenly divided between the machines, then there are \(O(n^4/m^3)\) possible choices to exchange the jobs. In fact, for each of the \(m(m - 1)\) couples of machines \((i, l)\), we have at most \([n/m]^4\) couples of two jobs \((j, k)\) and \((u, v)\). Note that Step 1 can be performed more times before \(JIP(w)\) stops.

The \(JIP1(w)\) and \(JIP2(w)\) procedures, that are slightly modified versions of \(JIP(w)\), attempt to lead the research process to regions of the solution space that have not been previously visited. Formally, the procedures can be described as follows.

**JIP1(w) Procedure**

\(JIP1(w)\) is obtained from \(JIP(w)\) by adding at the begin of Step 1: Sort the machines in non-increasing order with respect to their workloads and sort the elementary jobs belonging to each of the subsets \(S_i\) and \(S_j\) in non-increasing order with respect to their processing times.

**JIP2(w) Procedure**

\(JIP2(w)\) is obtained from \(JIP(w)\) by adding by adding at the begin of Step 1: Sort the machines in non-increasing order with respect to their workloads and sort the jobs belonging to \(S_i\) in non-increasing order with respect to their processing times and those belonging to \(S_j\) in non-decreasing order with respect to their processing times.
4. CONCLUSIONS

Minimizing workload imbalance across parallel resources is a goal of many production environments, yet there is no single measure of performance that has been accepted to characterize this practical goal. In this paper, three measures are described based on the statistical concept of dispersion. The three proposed criteria are described, including the need to normalize as a means to allow for experiments with diverse problem characterization (that is, number of machines, number of jobs, total processing load) to be analyzed properly. The paper presented an algorithm which uses several job exchange rules to maximize work balance. A logical extension of this paper is to test the proposed algorithm by simulation. Other areas of future research include the consideration of these three criteria for related parallel machine problems, particularly those with setups and batching machines.

REFERENCES


