SQU Journal for Science, 2020, 25(1), 26-47
Sultan Qaboos University

DOI:10.24200/squjs.vol25iss1pp26-47

Dimension Reduction and Relaxation of Johnson's Method for Two Machines Flow Shop Scheduling Problem

Mekonnen Redi* and Mohammad Ikram

Department of Mathematics, Arba Minch University, Arba Minch, Ethiopia.
**Email: mekonnen.redi@ambou.edu.et.*

ABSTRACT: The traditional dimensionality reduction methods can be generally classified into Feature Extraction (FE) and Feature Selection (FS) approaches. The classical FE algorithms are generally classified into linear and nonlinear algorithms. Linear algorithms such as Principal Component Analysis (PCA) aim to project high dimensional data to a lower-dimensional space by linear transformations according to certain criteria. The central idea of PCA is to reduce the dimensionality of the data set consisting of a large number of variables. In this paper, PCA was used to reduce the dimension of flow shop scheduling problems. This mathematical procedure transforms a number of (possibly) correlated jobs into a smaller number of uncorrelated jobs called principal components, which are the linear combinations of the original jobs. These jobs are carefully determined so that from the solution of the reduced problem multiple solutions of the original high dimensional problem can readily be obtained, or completely characterized, without actually listing the optimal solution(s). The results show that by fixing only some critical jobs at the beginnings and ends of the sequence using Johnson's method, the remaining jobs could be arranged in an arbitrary order in the remaining gap without violating the optimality condition that also guarantees minimum completion time. In this regard, Johnson's method was relaxed by terminating the listing of jobs at the first/last available positions when the job with minimum processing time on either machine attains the highest processing time on the other machine for the first time. By terminating Johnson's algorithm at an early stage, the method minimizes the time needed for sequencing those jobs that could be left arbitrarily. By allowing these jobs to be arranged in arbitrary order it gives job sequencing freedom for job operators without affecting minimum completion time. The results of the study were originally obtained for deterministic scheduling problems but they are more relevant on test problems randomly generated from uniform distribution $U[a,b]$ with lower bound a and upper bound b and normal distribution $N[\mu,\sigma^2]$ with mean μ and standard deviation σ^2.

Keywords: Dimension reduction; Flow shop scheduling problems; Principal component analysis; Relaxation of Johnson's algorithm.

تخفيض وإسترخاء البعد لطريقة جونسون لمسألة جدولة متجر التدفق في جهازين

ميكونين ريدي و محمد إكرام

الملخص: يمكن تصنيف طرق تخفيض الأبعاد التقليدية عمومًا بالاقتراب من استخراج الصفات (FE) واختيارها (FS). يتم تصنيف خوارزميات FE الكلاسيكية عمومًا في خوارزميات خطية وغير خطية. تهدف الخوارزميات الخطية مثل تحليل المكونات الرئيسية (PCA) إلى عرض البيانات عالية الأبعاد على فضاء أقل بعدًا باستخدام التحويلات الخطية وفقًا لمعايير معينة. إن فكرة PCA المركزية هي تخفيض الأبعاد لمجموعة البيانات التي تتكون من عدد كبير من المتحولات. لقد تم استخدام PCA في هذه الورقة لتخفيض البعد لمسائل جدولة متجر التدفق. يحول هذا الإجراء الرياضي عدد (ممكن) من الوظائف المرتبطة إلى عدد أصغر من الوظائف غير المرتبطة تسمى المكونات الرئيسية، وهي مجموعات خطية من الوظائف الأصلية. يتم تحديد هذه الوظائف بعناية حيث أن حل مسألة التخفيض يؤدي إلى حلول متعددة للمسألة الأصلية عالية الأبعاد، أو تمييزها بشكل كامل، بدون جدولة فعالية للحلول المثلى. تظهر النتائج أنه من خلال تثبيت عدد من الوظائف الحرجة فقط في بدايات ونهايات المتسلسلة باستخدام طريقة جونسون، أنه يمكن ترتيب الوظائف المتبقية بشكل عشوائي في الفجوة المتبقية دون انتهاك شرط الأمثلية الذي يضمن الحد الأدنى أيضا لاكتمال الزمن. وفي هذا الصدد، تم استرخاء طريقة جونسون بإنهاء قائمة الوظائف في المواضع المتاحة الأولى/الأخيرة عندما تكون الوظيفة مع زمن المعالجة الأدنى على أحد الأجهزة التي تصل لأعلى زمن للمعالجة على جهاز آخر لأول مرة.

من خلال إنهاء خوارزمية جونسون في مرحلة مبكرة، فإن الطريقة تقلل الزمن اللازم لتسلسل تلك الوظائف التي يمكن تركها بشكل عشوائي. من خلال السماح بترتيب هذه الوظائف عشوائيا يتم الحصول على حرية التسلسل لمؤثرات الوظيفة بدون التأثيرعلى الحد الأدنى لاكتمال الزمن. ففي الأصل تم الحصول على نتائج الدراسة لمسائل الجدولة الحتمية، لكنها كانت أكثر صلة بمسائل الاختبار التي تم إنشاؤها عشوائيًا من التوزيع المنتظم U[a,b] مع الحد الأدنى a والحد الأعلى b والتوزيع الطبيعي N[μ,σ^2] مع المتوسط μ والانحراف المعياري σ^2.

الكلمات المفتاحية: تخفيض البعد، مسائل جدولة متجر التدفق ، تحليل المكون الرئيسي ، إسترخاء خوارزمية جونسون.

1. Introduction

Two machines flow shop scheduling problem has been considered as a major problem in machine sequencing because it appears independently and as a sub-problem in the 'n-Jobs, m-Machines Problem'. The criterion of optimality in a flow shop scheduling problem is usually specified as minimization of makespan, which is defined as the time gap between the beginning of the first job on the first machine and finishing of the last job on the last machine to ensure that all jobs are completed on all machines. The objective of minimizing makespan in The Two Machines Flow Shop Scheduling model is also known as Johnson's problem.

The Johnson's algorithm is an exact solution method of the two machines, one-way, no-passing scheduling tasks problem, which serves as a basis for many heuristic algorithms. This rule is a complete list of ordering the jobs by filling the first or the last available space based on minimum operation times in the two machines from the waiting list into the optimal list until finally only one free space in the optimal list and one last job to be assigned remain in the waiting list. In this procedure, $n!$ comparisons are needed to obtain the optimal sequence. This needs large computation time for large size problems. To overcome this limitation, the current study identified a relaxation of Johnson's algorithm by developing an early stopover criteria, due to the fact that, after listing only some important jobs at the beginning and end of the optimal sequence using Johnson's method, it does not matter in whichever order the remaining jobs are operated as far as makespan is concerned. These criteria minimize the time needed for further computations of ordering the remaining jobs in either direction (first/last available positions) and it reduces the dimension of the problem. By allowing the remaining jobs to be arranged in an arbitrary order irrespective of Johnson's method without violating the optimality condition, the study creates job sequencing freedom for job operators to give priority without affecting minimum completion time.

2. Literature Review

Johnson [1] produced a pioneering work on machine sequencing literature and great advancements have been made in the field after other researchers also started to investigate solutions to many related problems. The studies discussed in this literature review are mainly concerned with relaxations that were oriented to produce alternate optimal solutions of the problem different from the strict Johnson's rule. In this regard, an early work on the relaxation of Johnson's method for The Two Machines Flow Shop Scheduling Problem was that of Ikram [2]. The study produced alternate ways of performing jobs in a way different from the one specified by Johnson's method without affecting minimum completion time by interchanging two jobs at a time from the optimal Johnson's sequence subject to the existence of certain conditions. In [3] also, the concept of [2] was used to find alternate optimal solutions for 'n-Jobs, 2-Machines Flow Shop Scheduling Problem with transportation time and equivalent job-block'. In [4] the idea was extended to 'n-Jobs, 3-Machines' Flow Shop Scheduling Problem' based on the work of [2]. These studies are indicators of increasing pressures on alternate ways of performing the same set of jobs in different ways without affecting minimum completion time. Alternate ways enable the assigning of priorities between jobs, and give freedom for job operators. For those groups of studies that generate alternate optimal sequences by interchanging two jobs at a time from the optimal Johnson's sequence, the total number of such alternate sequences is 2^k where k is the number of all interchangeable pairs.

In the original paper [1], Johnson also solved the 'n-Jobs, 3-Machines Flow Shop Scheduling Problem' in which the processing order for all the jobs in three machines A, B and C is $\mathrm{A} \rightarrow \mathrm{B} \rightarrow \mathrm{C}$ for two particular cases for which all jobs $\mathrm{j_i};\ \mathrm{i} = 1, 2, 3, \dots, \mathrm{n};$; satisfy $\max\{\mathrm{B}\} \leq \min\{\mathrm{A}\}$ or $\max\{\mathrm{B}\} \leq \min\{\mathrm{C}\}$. Few relaxations were made to this condition. The same assumption was made in [4] to the formulation of their alternate solution for the three machine problem. Conway, Maxwell and Miller [5] have shown that the same rule works if B is a non-bottleneck machine, i.e., is a machine that can process any number of jobs at the same time. Maggu, Alam and Ikram [6] also developed an algorithm for a special type of 'n -Jobs, m-Machines Scheduling Problem' which is an extension of Johnson's 'n-Jobs, m-Machines' sequencing rule.

The general 'n-Jobs, m-Machines Problem' becomes NP-complete [7] for all $m \geq 3$ (cannot be solved optimally in polynomial time) and the Johnson's algorithm can be applied only for some few particular cases that obey some primary conditions. The general 'n-Jobs, m-Machines' flow shop scheduling problems are NP-Hard, so exact optimization techniques are impractical for large size problems. In other words, classical optimization methods such as the branch and bound method, dynamic programming, etc. can be used only for small size problems. Problem size has been the main challenge for the development of solution methods for these problems because the solution space in its original form is of combinatorial order of number of jobs, which makes it more difficult to solve the problems in polynomial time for large n. Therefore, large size problems are solved by heuristic methods.

Many heuristic methods reduce the m machines into two virtual machines and apply Johnson's algorithm based on some specific rules or decisions. Important and earlier heuristic algorithms are due to Palmer [8], Gupta [9] and Campbell, Dudek, and Smith’s (CDS) heuristic [10]. The other heuristic is due to Nawaz, Enscore, and Ham [11], and is known as the NEH Heuristic. Both CDS and NEH are constructive heuristics. This means that they produce a sequence of at most $m-1$ solutions from which the best sequence is chosen. In these methods, all m machines are first divided into two groups which are considered as two virtual machines, and the problem is solved by applying Johnson's algorithm. The other heuristic, the HFC heuristic of Koulamas [12] on the other hand, is an improvement heuristic. In contrast, an improvement heuristic starts with a given sequence and searches for improvement, but the computational effort is unpredictable. Improvement heuristics are usually based on generic methods such as neighborhood search. They also use the algorithm of Johnson in the first phase, and then by specific rules, make better solutions, starting from an existing feasible solution in a sequence of steps.

In another heuristic, Rapid Access (RA) heuristic [13], two virtual machines are defined, and as in the CDS heuristic, method and weights are assigned, one for each virtual machine. Finally, the flow shop scheduling problem is solved by applying Johnson's algorithm. In [7] also, two variants of heuristic algorithms were developed to solve the classic flow shop scheduling problem. The first algorithm was a constructive heuristic, in which each job was placed in the optimal schedule based on a greedy-type selection. The second algorithm changes the construction of an optimal schedule in a stochastic manner. In [14] also, genetic algorithms were used to solve The Two Machines Flow Shop Problem with the objective of minimizing makespan.

The multi-objective flow shop scheduling problems have been the subject of extensive studies. The majority of bi-criteria flow shop investigations consider the combination of makespan with other performance measures [15-19]. In multi-objective problems, creating alternate solutions for the makespan objective by applying the results of this study will relax the other objectives and open more space for the applicability of multi-criteria decision making.

All the above works emphasize the importance of The Two Machines Flow Shop Scheduling Problem and the high reliance of solution methods on Johnson's algorithm. According to [7], even though the various studies have suggested many approaches, it is difficult to find the simplest approach to find an optimal sequence for solving the n-Jobs, m-Machines Flow Shop Scheduling Problems. Problem size has been the major challenge for all these heuristic methods. Researchers point out the need for scheduling algorithms to minimize makespan for the ‘n-Jobs m-Machines’ flow shop scheduling problems with the simplest steps as an alternative. However, little attempt has been made in the previous studies to decrease problem size for the applicability of the exact (heuristic) methods developed so far. Dimension reduction would create enough room for the application of these methods. Once the optimal sequence for the reduced problem is identified, it enables the complete characterization of all alternative optimal solutions of the original high dimensional problem. This research therefore considers dimension reduction for The Two Machine Flow Shop Scheduling Problem to be an important first step to decrease problem size, while at the same time creating alternative ways of sequencing jobs that guarantee the non-increase of total elapsed time on the fictitious machines formed by reducing the m machines into two virtual machines.

3. Problem Statement and Basic Assumptions

In this problem two machines (A and B) of high automation and unlimited buffer size are working together in such a way that machine A is always available to start the next job as soon as it finishes the current job. The finished jobs in machine A are then immediately transferred to the queue in machine B, and machine B operates the jobs in the same order as they have been executed in machine A. If there is no job in the queue, machine B has to wait until machine A finishes the current job. In this case an idle time occurs for machine B between its finishing of previous job and until machine A finishes its current one. The objective of the problem is therefore to minimize the sum of all these idle times in machine B for all the jobs from start to end. The criterion of optimality in a flow shop scheduling problem is usually specified as minimization of makespan, which is defined as the time gap between the beginning of the first job on the first machine and finishing of the last job on the last machine to ensure that all jobs are completed on both machines. Many variants of the problem have evolved since its formulation. As an objective function, mean flowtime, completion time variance and total tardiness can also be used.

The results originally obtained in [1] are among the very first formal results in the theory of scheduling. The objective of minimizing makespan in The Two Machines Flow Shop Scheduling model is also known as Johnson's problem. The Johnson's algorithm is an exact solution method of the two machines, one-way, no-passing scheduling tasks problem which serves as a basis for many heuristic algorithms [20]. The values of the processing times of a job j_i on machines A and B are denoted by a_{j_i} and b_{j_i} respectively for $i = 1, 2, \dots, n$ are deterministically known, constant and positive. They include also all the necessary auxiliary times involved in the technological process. The following important assumptions are made in the problem.

Assumption 1: A set of n unrelated, multiple-operation jobs are available for processing at time zero. (Each job requires 2 operations, and each operation requires a different machine.)

Assumption 2: Both machines are continuously available.

Assumption 3: Only one operation is carried out on a machine at a time.

Assumption 4: Once an operation begins in a machine, it proceeds without interruption.

Assumption 5: Processing times are known in advance and are deterministic, constant and positive.
Assumption 6: Setup times for the operations are sequence independent and are included in processing times.
Assumption 7: The time required in moving jobs from one machine to another is negligibly small.
Assumption 8: The same job-sequence is maintained over each machine, in other words no passing is allowed.

3.1. Algorithm: Johnson's Rule

Johnson's Rule: Job $j_i = (a_{j_i}, b_{j_i})$ precedes job $j_l = (a_{j_l}, b_{j_l})$ in an optimal sequence $\{j^*_i, i = 1, 2, 3, \dots, n\}$ if $min\{a_{j_i}, b_{j_l}\} \leq min\{a_{j_l}, b_{j_i}\}$.

In practice, an optimal sequence is directly constructed with an adaptation of Johnson's Rule. The positions in sequence are filled by a one-pass mechanism that, at each stage, identifies a job that should fill either the first [last] available position.

- *Step 1*

Examine the columns of A and B for processing times on machines A and B and find the smallest processing time among unscheduled jobs (waiting list).

- *Step 2a*

If the smallest processing time occurs for the first machine, then place the corresponding job in the first available position in the sequence (optimal list). (Ties may be broken arbitrarily.) Go to step 3.

- *Step 2b*

If the smallest processing time occurs for the second machine, then place the corresponding job in the last available position in the sequence (optimal list). (Ties may be broken arbitrarily.) Go to step 3.

- *Step 3*

Remove the assigned job from consideration (waiting list) and return to Step 1 until all sequence positions are filled.

3.2. Algorithm: Alternate formulation of Johnson's Rule

An alternative way to describe Johnson's Rule [21] that provides a different perspective on the structure of optimal schedules is used for this study. In this formulation, the problem is considered as a sequencing problem put mathematically as $P = \{j_i = (a_{j_i}, b_{j_i});\ i = 1, 2, 3, \dots, n\}$ where a_{j_i} and b_{j_i} are processing times of job j_i on machines A and B, respectively with no passing of jobs on the two machines in the order $A \rightarrow B$. The jobs in P are then partitioned into two disjoint clusters J_1 and J_2 where $J_1 = \{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} \leq b_{j_i};\ i = 1, 2, 3, \dots, n\}$ and $J_2 = \{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} > b_{j_i};\ i = 1, 2, 3, \dots, n\}$.

We call jobs in J_1 jobs of the first kind and jobs in J_2 jobs of the second kind. It is important to note that jobs in J_1 have longer processing time (at least equal) on the second machine while jobs in J_2 have strictly longer processing time on the first machine. Then in the optimal Johnson's sequence, jobs in J_1 are first arranged in a non-decreasing order of their processing times on the first machine and then jobs in J_2 are arranged in a non-increasing order of their processing times on the second machine. The present approach is often helpful and easy to apply and implement. Therefore, we follow this procedure to describe the optimal Johnson's sequence for The Two Machines Flow Shop Scheduling Problem.

- *Step 1:*

Let $J_1 = \{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} \leq b_{j_i};\ i = 1, 2, 3, \dots, n\}$ and $J_2 = \{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} > b_{j_i};\ i = 1, 2, 3, \dots, n\}$

- *Step 2:*

Arrange the members of set J_1 in non-decreasing order of a_{j_i} to get an ordered set J^*_1, and arrange the members of set J_2 in non-increasing order of b_{j_i} to get an ordered set J^*_2.

- *Step 3:*

An optimal sequence is the ordered set J^*_1 followed by the ordered set J^*_2.

Notation

The optimal Johnson's sequence for $P = \{j_i = (a_{j_i}, b_{j_i});\ i = 1, 2, 3, \dots, n\}$ is denoted by $\{j^*_i = (a_{j^*_i}, b_{j^*_i});\ i = 1, 2, 3, \dots, n\} = j^*_1 \rightarrow j^*_2 \rightarrow \cdots \rightarrow j^*_n$ with an asterisk (*) on j. We represent by $j^*_i = (a_{j^*_i}, b_{j^*_i})$ a job at the i^{th} position in an optimal Johnson's sequence with corresponding processing times on machines A and B, equal to $a_{j^*_i}$ units and $b_{j^*_i}$ units, respectively, and we denote its completion times on machines A and B by $C_1(j^*_i)$ and $C_{max}(j^*_i)$, respectively. We represent by $j_i = (a_{j_i}, b_{j_i})$ a job at the i^{th} position in a sub-optimal sequence with corresponding processing times on machines A and B, equal to a_{j_i} units and b_{j_i} , respectively, and its completion times on machines A and B by $C_1(j_i)$ and $C_{max}(j_i)$, respectively.

4. Data Mining and Dimension Reduction

Real-world data are typically noisy, enormous in volume, and may originate from a jumble of heterogenous sources [22]. Powerful and versatile tools are very much needed to automatically uncover valuable information from the tremendous amounts of data and to transform such data into organized knowledge. This necessity has led to the birth of data mining. Data mining is the process of discovering interesting patterns and knowledge from large amounts of data, having made a closer investigation of attributes and data values. As a general technology, data mining can be applied to any kind of data as long as the data are meaningful for a target application.

The state-of-art data mining tools could be employed for Flow Shop Scheduling Problems to overcome the limitations of the solution methods for large size problems. Due to the limited number of studies so far which have applied the state-of-the-art dimension reduction methods for Flow Shop Scheduling Problems, this paper examines The Two Machines Flow Shop Scheduling Problem more closely because of its significant contribution to Other Flow Shop Problems.

In the absence of real-world data for a typical study, generated data play an important role. For this study, generated data from a uniform distribution $U[a,b]$ with lower bound a and upper bound b and a normal distribution $N[\mu,\sigma^2]$ with mean μ and standard deviation σ^2 were used for different parameter values, and alternatives were analysed to describe a large size problem in terms of a small number of parameters. The effort was to discover important features of Flow Shop Scheduling Problems. This knowledge discovery process involves a sequence of logical understanding of the basic features of the high dimensional problem to extract a low dimensional representation of its key features.

As was suggested in [22], data mining involves a sequence of procedures that require the planners' understanding of the problem at hand. We further elaborate these steps in the context in which they have been applied to this study. Some of these procedures were applied in earlier studies of Flow Shop Scheduling Problems with a different context. For example, the first step, data integration (where multiple data items may be combined), was studied by equivalent job blocks. The concept of equivalent job blocking was introduced in [23] in the theory of scheduling. In the context of this study, however, multiple jobs were, for simplicity, represented by a single representative job, which may be different from the context of equivalent job blocks.

In the second step of data mining, data selection, data relevant to the analysis task are retrieved from the database. The other, following, steps of data mining were contextually used for Flow Shop Scheduling Problems to use them for the proposed study to fill the research gap in applying these techniques for dimension reduction.

In the third step, data transformation, data are transformed and consolidated into forms appropriate for mining. This step was used for this study to group jobs with identical processing times that hold equal priority in the optimal sequence. Thus it is enough to represent jobs with equal priority by a single representative job to reduce the dimension of the problem. This step was used in the principal component analysis (see Section 4. 1).

The fourth step, the data mining step, is an essential process where intelligent methods are applied to extract data patterns and to understand how only important patterns are exhibited. This step was used to develop mini-max criteria (see Section 4.2) to reduce the dimension of jobs of the first kind (see Section 4.2.1) and jobs of the second kind (see Section 4.2.2).

The fifth step, the pattern evaluation step, consists of identifying the truly interesting patterns representing knowledge based on interestingness measures, and checking whether further conclusions could be reached about the high dimensional problem from its low dimensional representation. This step was carried out in this research by formulating the results in the form of theorems and giving analytic mathematical proofs (see Section 4.2).

In the sixth step, the knowledge presentation step, illustrations are made to confirm the findings on a theory of knowledge; it aims to present mined knowledge convincingly to other users. This involves the organization of this study in a form of publication with all the necessary background information to acceptable competency levels. In particular, the illustration example in Section 5 also plays this role.

In the next sub-sections the main findings of the study are organized as theorems and algorithms. In section 4.1 Principal Component Analysis (PCA) is discussed in the sense of its application for dimension reduction of Flow Shop Scheduling Problems. In Section 4.2 further dimension reduction is carried out using mini-max criteria. Here, two investigations are made independently for the two kinds of jobs discussed earlier.

For jobs with more processing time on the second machine, an early stopover criteria was achieved when the job assigned to the first available position with minimum processing time criteria, for the first time, attains the highest processing time on the second machine for all jobs of the first kind (see Section 4.2.1). Similarly, for jobs with more processing time on the first machine, an early stopover criteria was achieved when the job assigned to the last available position with minimum processing time criteria, for the first time, attains the highest processing time on the first machine for all jobs of the second kind (see Section 4.2.2). Finally, these results are formulated in the form of algorithms. The first algorithm is a dimension reduction algorithm and it combines the dimension reductions carried out in Section 4.1 and Section 4.2 together. The second algorithm is a relaxation of Johnson's algorithm and it combines the early stopover criteria achieved in Section 4.1 and Section 4.2 as termination criteria for Johnson's algorithm.

4.1. Dimension Reduction by Principal Component Analysis

The state-of-the-art Dimensional Reduction (DR) methods are divided into projective methods and methods that model the manifold on which the data lies. Perhaps the simplest approach is to attempt to find low dimensional projections that extract useful information from the data, by maximizing a suitable objective function [25]. Cluster analysis is one of the major data analysis methods which are widely used for many practical applications. The purpose of clustering is to group together data points, which are close to one another [26].

Let problem $P = \{j_i = (a_{j_i}, b_{j_i});\ i = 1, 2, 3, \dots, n\}$ be the original high dimensional problem where the notation $j_i = (a_{j_i}, b_{j_i});\ i = 1, 2, 3, \dots, n$ means job j_i has processing time equal to a_{j_i} units on machines A and processing time of b_{j_i} units on machine B , and the aim is to find the optimal sequence $S = \{{j^*}_i;\ i = 1, 2, 3, \dots, n\}$ that minimizes total elapsed time of operating all jobs in the same order in the two machines uninterrupted with no passing of jobs on the two machines in the order $A \rightarrow B$.

Now we partition all jobs in P into two clusters, by defining a function $J: P \rightarrow \{0,1\}$ given by the formula (1).

$$J(j_i) = J\left((a_{j_i}, b_{j_i})\right) = \begin{cases} 0\ if\ j_i = (a_{j_i}, b_{j_i})\ \in P: a_{j_i} \le b_{j_i} \\ 1\ if\ j_i = (a_{j_i}, b_{j_i})\ \in P: a_{j_i} > b_{j_i} \end{cases} \tag{1}$$

Let
$J_1 = \left\{j_i = (a_{j_i}, b_{j_i})\ \in P: J(j_i) = J\left((a_{j_i}, b_{j_i})\right) = 0\right\}$ and $J_2 = \left\{j_i = (a_{j_i}, b_{j_i})\ \in P: J(j_i) = J\left((a_{j_i}, b_{j_i})\right) = 1\right\}$.

Define the projection mapping : $f: P \rightarrow \{0,1\} \times \Re$ given by the formula (2).

$$f(j_i) = f\left((a_{j_i}, b_{j_i})\right) = \begin{cases} (0, a_{j_i})\ if\ J\left((a_{j_i}, b_{j_i})\right) = 0 \\ (1, b_{j_i})\ if\ J\left((a_{j_i}, b_{j_i})\right) = 1 \end{cases} \tag{2}$$

This map specifies the kind of job and the corresponding processing time of the job that is relevant for assigning it to the optimal Johnson's sequence.
Let the image of P under f be given by $Im(P) = f(P) = \{x: x = f(j_i)\ for\ some\ j_i \in P\}$. We call the elements of $f(P)$ Principal Components of P.
Then the inverse image of f is given by $f^{-1}(P): f(P) \rightarrow P$ given by the formula (3).

$$f^{-1}(x) = \left\{j_i = (a_{j_i}, b_{j_i})\ \in P: f(j_i) = f\left((a_{j_i}, b_{j_i})\right) = x\right\} \tag{3}$$

Equation (3) clusters all jobs into disjoint equivalence classes consisting of exactly those jobs in J_1 that have equal processing times on the first machine or those jobs in J_2 that have equal processing times on the second machine. In the optimal Johnson's sequence the jobs in each cluster are arranged successively. Let there be K principal components $x_1, x_2, \dots, x_K$ with $\eta_1, \eta_2, \dots, \eta_K$ number of jobs in $f^{-1}(x_1), f^{-1}(x_2), \dots, f^{-1}(x_K)$, respectively, such that $\eta_1 + \eta_2 + \dots + \eta_K = n$.

Thus all jobs in the same cluster could be represented by a single job from the group and number of reserved positions for these jobs by $\eta_1, \eta_2, \dots, \eta_K$. In particular if we determine to represent each cluster with the job that has the longest total processing time on the two machines, then a unique identifier is assigned to each job.
Define $f^*: \{x_1, x_2, \dots, x_K\} \rightarrow P$ given by the formula (4).

$$f^*(x_k) = j_k = (a_{j_k}, b_{j_k}) \text{ where } a_{j_k} + b_{j_k} = max_{(a_{j_i}, b_{j_i}) \in f^{-1}(x_k)}\{a_{j_i} + b_{j_i}\} \tag{4}$$

This is a choice function selecting the job with highest total processing time on the two machines from each cluster. The collection $P^* = \left\{\left(f^*(x_k), \eta_k, f^{-1}(x_k)\right), k = 1, 2, \dots, K\right\}$ characterizes all jobs in P that occupy equal priority in the optimal Johnson's sequence by $f^{-1}(x_k)$, the total number of jobs in the group by η_k and a representative job $f^*(x_k)$ for $k = 1, 2, \dots, K$. This way the original high dimensional problem is represented by a lower dimensional sub-problem P^*. For each job $j_i = (a_{j_i}, b_{j_i}) \in P$, there is a unique identifier $\left(f^*(x_k), \eta_k, f^{-1}(x_k)\right) \in P^*$ such that $f\left((a_{j_i}, b_{j_i})\right) = f\left((a_{j_k}, b_{j_k})\right)$ and $f^*(x_k) = (a_{j_k}, b_{j_k})$.

Thus, the original high dimensional Flow Shop Scheduling Problem was defined in terms of the low dimensional sub-problem from which the solution of the original Two Machines Flow Shop Scheduling Problem could be obtained more readily by replacing the representative job by the block of jobs it corresponds to. More specifically, the original 'n -Job 2-Machine Problem' is reduced into K clusters with $\eta_1, \eta_2, \dots, \eta_K$ number of jobs respectively where $\eta_1 + \eta_2 + \cdots + \eta_K = n$. At this stage of dimension reduction there are at least $\eta_1! * \eta_2! * \dots * \eta_K!$ alternative optimal sequences to this problem.

4.2. Dimension Reduction of Jobs by Means of Mini-Max Criteria

The traditional and the state-of-the-art dimension reduction methods can be generally classified into Feature Extraction (FE) and Feature Selection (FS) approaches. FE algorithms aim to extract features by projecting the original high-dimensional data to a lower-dimensional space through algebraic transformations [26]. The classical FE algorithms are generally classified into linear and nonlinear approaches. In contrast to the FE algorithms, FS algorithms have been widely used on large-scale data and aim at finding out a subset of the most representative features according to some objective function. It is optional that we assume dimensional reduction by PCA method discussed in the previous section has already been carried out for the problem before we apply Mini-Max Criteria in this section.

Principle of Mathematical Induction (PMI)

We use the Principle of Mathematical Induction (PMI) to prove the results of the next section. It states as follows:
Let $P(n)$ be a property that depends on natural numbers n satisfies the following two conditions:
i. $P(k_0)$ holds true and
ii. $P(k+1)$ holds true whenever $P(k)$ holds true.
Then $P(n)$ holds true for all natural numbers $n \geq k_0$.

4.2.1. Dimension Reduction of Jobs with More Processing Time on the Second Machine

Theorem 1

Let the 'n- Jobs, 2- Machines' sequencing problem $P = \{j_i = (a_{j_i}, b_{j_i});\ i = 1, 2, 3, \dots, n\}$ be in the reduced form after dimension reduction using PCA has been performed where the notation $j_i = (a_{j_i}, b_{j_i})$ means a_{j_i} and b_{j_i} are processing times of job j_i on machines A and B respectively for all $i = 1, 2, 3, \dots, n$ and the machine order is $A \to B$ with no passing rule. Suppose we partition all jobs in P into two disjoint classes J_1 and J_2 where $J_1 = \{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} \leq b_{j_i};\ i = 1, 2, 3, \dots, n\}$ and $J_2 = \{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} > b_{j_i};\ i = 1, 2, 3, \dots, n\}$. Let there be n_1 jobs in J_1 and n_2 jobs in J_2 with $n_1 + n_2 = n$. Let the sequence $\{j^*_i; i = 1, 2, 3, \dots, n\} = j^*_1 \to j^*_2 \to j^*_3 \to \cdots \to j^*_n$ be an optimal Johnson's sequence obtained by the second alternative rule. Consider the jobs in J_1 only. Let $b_{j^*_k} = max_{a_{j_i} \leq b_{j_i}}\{b_{j_i}: (a_{j_i}, b_{j_i}) \in P\} = max_{(a_{j_i}, b_{j_i}) \in J_1}\{b_{j_i}\}$ occur in the optimal sequence for some job $j^*_k = (a_{j^*_k}, b_{j^*_k})$ where $1 \leq k \leq n_1$. If this job is not unique, choose the first occurrence of all such jobs. That is, choose the job $j^*_k = (a_{j^*_k}, b_{j^*_k})$ such that $b_{j^*_k} = max_{(a_{j_i}, b_{j_i}) \in J_1}\{b_{j_i}\}$ and $a_{j^*_k} = min_{b_{j_i = b_{j^*_k}}}\{a_{j_i}: (a_{j_i}, b_{j_i}) \in J_1\}$. Then the block of jobs $j^*_{k+1} \to j^*_{k+2} \to j^*_{k+3} \to \cdots \to j^*_{n_1}$ could be arranged in an arbitrary order within the block while keeping all the remaining jobs fixed in their position in the optimal Johnson's sequence without violating the optimality condition. Moreover, these jobs do not create any idle time for machine , and the completion time of job j^*_{k+m} on machine B is given by the formula (5) for $k \leq k + m \leq n_1$.

$$C_{max}\left(j^*_{k+m}\right) = C_{max}\left(j^*_k\right) + \sum_{i=1}^{m} b_{j^*_{k+i}} \tag{5}$$

Consequently, the completion time of all jobs of the first kind on machine B is given by the formula (6) irrespective of the order of operation of the jobs $j^*_{k+1} \to j^*_{k+2} \to j^*_{k+3} \to \cdots \to j^*_{n_1}$.

$$C_{max}\left(j^*_{n_1}\right) = C_{max}\left(j^*_k\right) + \sum_{i=k+1}^{n_1} b_{j^*_i} \tag{6}$$

Proof

Let $\{j^*_i; i = 1, 2, 3, \dots, n\} = j^*_1 \to j^*_2 \to j^*_3 \to \cdots \to j^*_n$ be an optimal Johnson's sequence, and j^*_k be the job described in the theorem.
Let machine A finish job j^*_k at $C_1\left(j^*_k\right)$ and machine B finish job j^*_k at $C_{max}\left(j^*_k\right)$ and let the sequence of jobs $j_{k+1} \to j_{k+2} \to j_{k+3} \to \cdots \to j_{n_1}$ be any permutation of the sequence $j^*_{k+1} \to j^*_{k+2} \to j^*_{k+3} \to \cdots \to j^*_{n_1}$.

For any job $j_i; i = 1, 2, 3, \dots, n$ the following relation (7) holds.

$$C_1(j_i) + b_{j_i} \le C_{max}(j_i) \tag{7}$$

from which we get also $C_1({j^*}_k) + b_{{j^*}_k} \le C_{max}({j^*}_k)$.

For any job $j_{k+i}; i = 1, 2, \dots, n_1 - k$ the following relations (8)-(9) hold.

$$a_{j_{k+i}} \le b_{j_{k+i}} \le b_{{j^*}_k} \tag{8}$$
$$a_{{j^*}_k} \le a_{j_{k+i}} \le b_{j_{k+i}} \tag{9}$$

Let us consider the starting and finishing times of the jobs in $\{j_{k+i}; i = 1, 2, \dots, n_1 - k\}$ on the two machines A and B. It is important to note that at any stage of machine sequencing, if machine B finishes job j_i at $t = C_{max}(j_i)$ and machine A finishes the next job j_{i+1} at $t = C_1(j_{i+1})$, then machine B starts job j_{i+1} at $max\{C_1(j_{i+1}), C_{max}(j_i)\}$.

1. Machine A starts job j_{k+1} at $t = C_1({j^*}_k)$ and completes it at $t = C_1(j_{k+1}) = C_1({j^*}_k) + a_{j_{k+1}}$. Using equations (7)-(9), the finishing time of job j_{k+1} on machine A satisfies the relations (10)-(13) below.

$$C_1(j_{k+1}) = C_1({j^*}_k) + a_{j_{k+1}} \tag{10}$$
$$C_1(j_{k+1}) \le C_1({j^*}_k) + b_{j_{k+1}} \tag{11}$$
$$C_1(j_{k+1}) \le C_1({j^*}_k) + b_{{j^*}_k} \tag{12}$$
$$C_1(j_{k+1}). \le C_{max}({j^*}_k) \tag{13}$$

The right side of (13) is the finishing time of job ${j^*}_k$ on machine B. Hence machine B starts job j_{k+1} at $max\{C_1(j_{k+1}), C_{max}({j^*}_k)\} = C_{max}({j^*}_k)$. Thus, this job does not create any idle time for machine B. The finishing time of this job on machine B is given by equation (14).

$$C_{max}(j_{k+1}) = C_{max}({j^*}_k) + b_{j_{k+1}} = C_{max}({j^*}_k) + \textstyle\sum_{i=1}^{1} b_{j_{k+i}} \tag{14}$$

Thus, the theorem holds true for $m = 1$.

2. Machine A starts job j_{k+2} at $t = C_1(j_{k+1})$ and completes it at $t = C_1(j_{k+2}) = C_1(j_{k+1}) + a_{j_{k+2}}$. Using equations (7)-(9), the finishing time of job j_{k+2} on machine A satisfies the relations (15)-(18) below.

$$C_1(j_{k+2}) = C_1({j^*}_k) + a_{j_{k+1}} + a_{j_{k+2}} \tag{15}$$
$$C_1(j_{k+2}) \le C_1({j^*}_k) + b_{j_{k+1}} + b_{j_{k+2}} \tag{16}$$
$$C_1(j_{k+2}) \le C_1({j^*}_k) + b_{j_{k+1}} + b_{{j^*}_k} \tag{17}$$
$$C_1(j_{k+2}) \le C_1({j^*}_k) + b_{{j^*}_k} + b_{j_{k+1}} \tag{18}$$
$$C_1(j_{k+2}) \le C_{max}({j^*}_k) + b_{j_{k+1}} = C_{max}(j_{k+1}) \tag{19}$$

The right side of (19) is the finishing time of job j_{k+1} on machine B. Hence machine B starts job j_{k+2} at $max\{C_1(j_{k+2}), C_{max}(j_{k+1})\} = C_{max}(j_{k+1})$. Thus, this job does not create any idle time for machine B. Using (14) the finishing time of this job on machine B is given by equations (20)-(21).

$$C_{max}(j_{k+2}) = C_{max}(j_{k+1}) + b_{j_{k+2}} = C_{max}({j^*}_k) + b_{j_{k+1}} + b_{j_{k+2}} \tag{20}$$
$$C_{max}(j_{k+2}) = C_{max}(j_{k+1}) + b_{j_{k+2}} = C_{max}({j^*}_k) + \textstyle\sum_{i=1}^{2} b_{j_{k+i}} \tag{21}$$

Thus, the theorem holds true for $m = 2$.

Induction assumption

3. Suppose the formula works for m such that $k+1 \le k+m \le n_1$ i. e. machine B finishes job $j_{k+\mathrm{m}}$ at formula (22).

$$C_{max}\left(j_{k+m}\right) = C_{max}\left({j^*}_k\right) + b_{j_{k+1}} + b_{j_{k+2}} + \cdots + b_{j_{k+m}} \tag{22}$$

$$C_{max}\left(j_{k+m}\right) = C_{max}\left({j^*}_k\right) + \sum_{i=1}^{m} b_{j_{k+i}} \tag{23}$$

Then machine A finishes job $j_{k+\mathrm{m}+1}$ at time $t = C_1\left(j_{k+m+1}\right)$ given by equation (24)

$$C_1\left(j_{k+m+1}\right) = C_1\left({j^*}_k\right) + a_{j_{k+1}} + a_{j_{k+2}} + \cdots + a_{j_{k+m+1}} \tag{24}$$

Using equations (7)-(9), the finishing time of job $j_{k+\mathrm{m}+1}$ on machine A satisfies the relations (25)-(29) below.

$$C_1\left(j_{k+\mathrm{m}+1}\right) = C_1\left({j^*}_k\right) + a_{j_{k+1}} + a_{j_{k+2}} + \cdots + a_{j_{k+\mathrm{m}+1}} \tag{25}$$

$$C_1\left(j_{k+\mathrm{m}+1}\right) \le C_1\left({j^*}_k\right) + b_{j_{k+1}} + b_{j_{k+2}} + \cdots + b_{j_{k+\mathrm{m}+1}} \tag{26}$$

$$C_1\left(j_{k+\mathrm{m}+1}\right) \le C_1\left({j^*}_k\right) + b_{j_{k+1}} + b_{j_{k+2}} + \cdots + b_{j_{k+\mathrm{m}}} + b_{{j^*}_k} \tag{27}$$

$$C_1\left(j_{k+\mathrm{m}+1}\right) \le C_1\left({j^*}_k\right) + b_{{j^*}_k} + b_{j_{k+1}} + b_{j_{k+2}} + \cdots + b_{j_{k+\mathrm{m}}} \tag{28}$$

$$C_1\left(j_{k+\mathrm{m}+1}\right) \le C_{max}\left({j^*}_k\right) + b_{j_{k+1}} + b_{j_{k+2}} + \cdots + b_{j_{k+\mathrm{m}}} = C_{\max}\left(j_{k+\mathrm{m}}\right) \tag{29}$$

The right side of (29) is the finishing time of job $j_{k+\mathrm{m}}$ on machine B. Hence machine B starts job $j_{k+\mathrm{m}+1}$ at $max\left\{C_1\left(j_{k+m+1}\right), C_{max}\left(j_{k+m}\right)\right\} = C_{max}\left(j_{k+m}\right)$. Thus, this job does not create any idle time for machine B. The finishing time of this job on machine B is given by equations (30)-(31).

$$C_{max}\left(j_{k+m+1}\right) = C_{max}\left({j^*}_k\right) + b_{j_{k+1}} + b_{j_{k+2}} + \cdots + b_{j_{k+m+1}} \tag{30}$$

$$C_{max}\left(j_{k+m+1}\right) = C_{max}\left({j^*}_k\right) + \sum_{i=1}^{m+1} b_{j_{k+i}} \tag{31}$$

Hence the formula also works for $m+1$. Thus, by the principle of mathematical induction, formula (5) holds for all m such that $k+1 \le k+m \le n_1$.

Thus, the finishing time of job ${j^*}_{n_1}$ is given by:

$$C_{max}\left({j^*}_{n_1}\right) = C_{max}\left({j^*}_k\right) + b_{{j^*}_{k+1}} + b_{{j^*}_{k+2}} + \cdots + b_{{j^*}_{n_1}} \tag{32}$$

$$C_{max}\left({j^*}_{n_1}\right) = C_{max}\left({j^*}_k\right) + \sum_{i=k+1}^{n_1} b_{{j^*}_i} \tag{33}$$

Equation (33) is identical to (6).
The total elapsed time to finish n_1 jobs of the first kind on machine B is $C_{max}\left({j^*}_k\right) + b_{{j^*}_{k+1}} + b_{{j^*}_{k+2}} + \cdots + b_{{j^*}_{n_1}}$ and it is independent of the order of the jobs $j_{k+1}, j_{k+2}, j_{k+3}, \ldots, j_{n_1}$ and it creates no additional idle time for machine B after job ${j^*}_k$. Since the jobs $j_{k+1}, j_{k+2}, j_{k+3}, \ldots, j_{n_1}$ are permutations of the jobs ${j^*}_{k+1}, {j^*}_{k+2}, {j^*}_{k+3}, \ldots, {j^*}_{n_1}$, we get $C_{max}\left({j^*}_k\right) + b_{j_{k+1}} + b_{j_{k+2}} + \cdots + b_{j_{n_1}} = C_{max}\left({j^*}_k\right) + b_{{j^*}_{k+1}} + b_{{j^*}_{k+2}} + \cdots + b_{{j^*}_{n_1}}$. The optimal sequence is also obtained by one of these permutations and the total elapsed time in the optimal sequence to finish n_1 jobs of the first kind is $C_{max}\left({j^*}_k\right) + b_{{j^*}_{k+1}} + b_{{j^*}_{k+2}} + \cdots + b_{{j^*}_{n_1}}$. If, further, we keep all jobs in J_2 fixed in the optimal sequence, the total elapsed time to finish all jobs with the above freedom will be equal to the optimal value. Hence any sequence obtained by the above sequencing freedom is optimal. Since we are free to arrange the jobs ${j^*}_{k+1}, {j^*}_{k+2}, {j^*}_{k+3}, \ldots, {j^*}_{n_1}$ in an arbitrary order, the above machine sequencing freedom creates $(n_1 - k)!$ sequences all of which are optimal.
For references, we call the job ${j^*}_k$ in the above theorem formulation the minimal job of $\left\{{j^*}_i; i = 1, 2, 3, \ldots, n\right\} = {j^*}_1 \to {j^*}_2 \to {j^*}_3 \to \cdots \to {j^*}_n$ and we call the block of jobs ${j^*}_{k+1} \to {j^*}_{k+2} \to {j^*}_{k+3} \to \ldots \to {j^*}_{n_1}$ the free jobs of first kind.

4.2.2. Dimension Reduction of Jobs with More Processing Time on the First Machine

Theorem 2

Let the 'n- Jobs, 2- Machines' sequencing problem $P = \{j_i = (a_{j_i}, b_{j_i});\ i = 1,2,3,\dots,n\}$ be in the reduced form after dimension reduction using PCA has been performed where the notation $j_i = (a_{j_i}, b_{j_i})$ means a_{j_i} and b_{j_i} are processing times of job j_i on machines A and B respectively for all $i = 1,2,3,\dots,n$ and the machine order is $A \to B$ with no passing rule. Suppose we partition all jobs in P into two disjoint classes J_1 and J_2 where $J_1 = \{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} \le b_{j_i};\ i = 1,2,3,\dots,n\}$ and $J_2 = \{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} > b_{j_i};\ i = 1,2,3,\dots,n\}$. Let there be n_1 jobs in J_1 and n_2 jobs in J_2 with $n_1 + n_2 = n$. Let the sequence $\{j^*_i; i = 1,2,3,\dots,n\} = j^*_1 \to j^*_2 \to j^*_3 \to \cdots \to j^*_n$ be an optimal Johnson's sequence obtained by the second alternative rule. Consider the jobs in J_2 only. Let $a_{j^*_q} = \max_{a_{j_i} > b_{j_i}} \{a_{j_i}: (a_{j_i}, b_{j_i}) \in P\} = \max_{(a_{j_i}, b_{j_i}) \in J_2}\{a_{j_i}\}$ occur in the optimal sequence for some job $j^*_q = (a_{j^*_k}, b_{j^*_k})$ where $n_1 + 1 \le q \le n$. If this job is not unique, choose the last occurrence of all such jobs. That is, choose the job $j^*_q = (a_{j^*_q}, b_{j^*_q})$ such that $a_{j^*_q} = \max_{(a_{j_i}, b_{j_i}) \in J_2}\{a_{j_i}\}$ and $b_{j^*_q} = \min_{a_{j_i} = a_{j^*_q}} \{b_{j_i}, (a_{j_i}, b_{j_i}) \in J_2\}$. Let the sequence $\{j_i; i = n_1 + 1, \dots, q-2, q-1\} = j_{n_1+1} \to \cdots \to j_{q-2} \to j_{q-1}$ be any permutation of the sequence $\{j^*_i; i = n_1 + 1, \dots, q-2, q-1\} = j^*_{n_1+1} \to \cdots \to j^*_{q-2} \to j^*_{q-1}$. Then individual jobs in the block of jobs $j^*_{n_1+1} \to \cdots \to j^*_{q-2} \to j^*_{q-1}$ could be arranged in an arbitrary order among themselves while keeping all the remaining jobs fixed in their position in the optimal sequence without violating the optimality condition. In other words, the sequence $j^*_1 \to j^*_2 \to j^*_3 \to \cdots \to j^*_{n_1} \to j_{n_1+1} \to \cdots \to j_{q-2} \to j_{q-1} \to j^*_q \to j^*_{q+1} \to j^*_{q+2} \to j^*_{q+3} \to \cdots \to j^*_n$ is also optimal.

More specifically, the completion time of the job j^*_q on machine B is independent of the order of jobs $j_{n_1+1}, \dots, j_{q-2}, j_{q-1}$ and is given by equation (34).

$$C_{max}\left(j^*_q\right) = max\ \left\{C_1\left(j^*_{q-m-1}\right) + a_{j^*_q} + b_{j^*_q} + \textstyle\sum_{i=1}^m a_{j_{q-i}}, C_{max}\left(j^*_{q-m-1}\right) + b_{j^*_q} + \textstyle\sum_{i=1}^m b_{j_{q-i}}\right\}.$$

Proof

Let the sequence $j_{n_1+1} \to \cdots \to j_{q-2} \to j_{q-1}$ be any permutation of the sequence $j^*_{n_1+1} \to \cdots \to j^*_{q-2} \to j^*_{q-1}$. Let us consider the finishing times of the job j^*_q starting from any job $j_{n_1+1}, \dots, j_{q-2}, j_{q-1}$ downward.

First, observe that for all jobs $j_{n_1+1}, \dots, j_{q-2}, j_{q-1}$ we see that the following relations (35) - (36) hold.

$$b_{j_{q-i}} < a_{j_{q-i}} \le a_{j^*_q} \tag{35}$$

$$b_{j^*_q} \le b_{j_{q-i}} \le a_{j_{q-i}} \tag{36}$$

We use induction on the number of jobs in between n_1 and q.

1. We first show the theorem holds for the case when there is no job in between n_1 and q i. e. $q - 1 = n_1$. In this case, machine A starts job j^*_q at $C_1\left(j^*_{n_1}\right)$ and completes it at $C_1\left(j^*_q\right)$ given by equation (37).

$$C_1\left(j^*_q\right) = C_1\left(j^*_{n_1}\right) + a_{j^*_q} \tag{37}$$

Thus machine B starts job j^*_q at $max\ \left\{C_1\left(j^*_{n_1}\right) + a_{j^*_q}, C_{max}\left(j^*_{n_1}\right)\right\}$, and completes it at $C_{max}\left(j^*_q\right)$ given by equation (38)

$$C_{max}\left(j^*_q\right) = max\ \left\{C_1\left(j^*_{n_1}\right) + a_{j^*_q} + b_{j^*_q}, C_{max}\left(j^*_{n_1}\right) + b_{j^*_q}\right\} \tag{38}$$

If we replace $m = 0$ in the formula in equation (34), we get equations (39)-(41).

$$max\ \left\{C_1\left(j^*_{q-0-1}\right) + a_{j^*_q} + b_{j^*_q} + \textstyle\sum_{i=1}^0 a_{j_{q-i}}, C_{max}\left(j^*_{q-0-1}\right) + b_{j^*_q} + \textstyle\sum_{i=1}^0 b_{j_{q-i}}\right\} \tag{39}$$

$$= max\ \left\{C_1\left(j^*_{q-1}\right) + a_{j^*_q} + b_{j^*_q}, C_{max}\left(j^*_{q-1}\right) + b_{j^*_q}\right\} \tag{40}$$

$$= max\ \left\{C_1\left(j^*_{n_1}\right) + a_{j^*_q} + b_{j^*_q}, C_{max}\left(j^*_{n_1}\right) + b_{j^*_q}\right\} \tag{41}$$

Equation (41) is identical to equation (38).

Thus, the theorem holds true for the case when $q-1=n_1$.

2. We next show the theorem holds for the case when there is only one job in between n_1 and q i. e. $q-2=n_1$. In this case, machine A starts job j_{n_1+1} at $C_1\left({j^*}_{n_1}\right)$ and completes it at $C_1\left(j_{n_1+1}\right) = C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}$ and machine B starts job j_{n_1+1} at $max\left\{C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}, C_{max}\left({j^*}_{n_1}\right)\right\}$, and completes it at $C_{max}\left(j_{n_1+1}\right)=max\left\{C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}+b_{j_{n_1+1}}, C_{max}\left({j^*}_{n_1}\right)+b_{j_{n_1+1}}\right\}$.

Then, machine B starts job ${j^*}_q$ at $C_1\left(j_{n_1+1}\right)$ and completes it at $C_1\left({j^*}_q\right)=C_1\left(j_{n_1+1}\right)+a_{{j^*}_q}$ and machine B starts job ${j^*}_q$ at $C_{max}\left({j^*}_q\right)=max\left\{C_1\left({j^*}_q\right), C_{max}\left(j_{n_1+1}\right)\right\}$.

$$max\left\{C_1\left(j_{n_1+1}\right)+a_{{j^*}_q}, max\left\{C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}+b_{j_{n_1+1}}, C_{max}\left({j^*}_{n_1}\right)+b_{j_{n_1+1}}\right\}\right\} \quad (42)$$

$$max\left\{C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}+a_{{j^*}_q}, C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}+b_{j_{n_1+1}}, C_{max}\left({j^*}_{n_1}\right)+b_{j_{n_1+1}}\right\} \quad (43)$$

But since $b_{j_{n_1+1}} \le a_{j_{n_1+1}} \le a_{{j^*}_q}$, we get $C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}+b_{j_{n_1+1}} \le C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}+a_{{j^*}_q}$. Thus, the maximum of the three numbers in equation (43) reduces to a maximum of two numbers in equation (44).

$$max\left\{C_1\left({j^*}_{n_1}\right)+a_{j_{n_1+1}}+a_{{j^*}_q}, C_{max}\left({j^*}_{n_1}\right)+b_{j_{n_1+1}}\right\} \quad (44)$$

Thus, the finishing time of job ${j^*}_q$ on machine B is given by the equation (45)

$$C_{max}\left({j^*}_q\right)=max\left\{C_1\left({j^*}_{n_1}\right)+a_{{j^*}_q}+b_{{j^*}_q}+a_{j_{n_1+1}}, C_{max}\left({j^*}_{n_1}\right)+b_{{j^*}_q}+b_{j_{n_1+1}}\right\} \quad (45)$$

If we put $m=1$, in the formula (34) we get (46)-(49)

$$C_{max}\left({j^*}_q\right)=max\left\{C_1\left({j^*}_{q-1-1}\right)+a_{{j^*}_q}+b_{{j^*}_q}+\textstyle\sum_{i=1}^{1} a_{j_{q-i}}, C_{max}\left({j^*}_{q-1-1}\right)+b_{{j^*}_q}+\textstyle\sum_{i=1}^{1} b_{j_{q-i}}\right\}$$

$$C_{max}\left({j^*}_q\right)=max\left\{C_1\left({j^*}_{q-2}\right)+a_{{j^*}_q}+b_{{j^*}_q}+a_{j_{q-1}}, C_{max}\left({j^*}_{q-2}\right)+b_{{j^*}_q}+b_{j_{q-1}}\right\} \quad (47)$$

$$C_{max}\left({j^*}_q\right)=max\left\{C_1\left({j^*}_{n_1}\right)+a_{{j^*}_q}+b_{{j^*}_q}+a_{j_{q-1}}, C_{max}\left({j^*}_{n_1}\right)+b_{{j^*}_q}+b_{j_{q-1}}\right\} \quad (48)$$

$$C_{max}\left({j^*}_q\right)=max\left\{C_1\left({j^*}_{n_1}\right)+a_{{j^*}_q}+b_{{j^*}_q}+a_{j_{n_1+1}}, C_{max}\left({j^*}_{n_1}\right)+b_{{j^*}_q}+b_{j_{n_1+1}}\right\} \quad (49)$$

Equation (49) is identical to equation (45).

Thus, the formula works for $m=1$.
Thus, the theorem holds for the case when $q-2=n_1$.

Induction assumption

3. Suppose for any m such that $n_1+1 \le q-m \le q$, the finishing time of the job ${j^*}_q$ on machine B is independent of the order of jobs $j_{q-m}, \dots, j_{q-2}, j_{q-1}$ and is given by equation (50).

$$C_{max}\left({j^*}_q\right)=max\left\{C_1\left({j^*}_{q-m-1}\right)+a_{{j^*}_q}+b_{{j^*}_q}+\textstyle\sum_{i=1}^{m} a_{j_{q-i}}, C_{max}\left({j^*}_{q-m-1}\right)+b_{{j^*}_q}+\textstyle\sum_{i=1}^{m} b_{j_{q-i}}\right\}$$

We want to show that the formula works for $m+1$. It is sufficient that we show the finishing time of the job ${j^*}_q$ on machine B is independent of the order of jobs $j_{q-m-1}, j_{q-m}, \dots, j_{q-2}, j_{q-1}$ and is given by equation (51).

$$C_{max}\left({j^*}_q\right)=max\left\{C_1\left({j^*}_{q-m-2}\right)+a_{{j^*}_q}+b_{{j^*}_q}+\textstyle\sum_{i=1}^{m+1} a_{j_{q-i}}, C_{max}\left({j^*}_{q-m-2}\right)+b_{{j^*}_q}+\textstyle\sum_{i=1}^{m+1} b_{j_{q-i}}\right\} \quad (51)$$

Machine A starts job j_{q-m-1} at $C_1\left({j^*}_{q-m-2}\right)$ and completes it at $C_1\left(j_{q-m-1}\right)=C_1\left({j^*}_{q-m-2}\right)+a_{j_{q-m-1}}$ and machine B starts job j_{q-m-1} at $max\left\{C_1\left({j^*}_{q-m-2}\right)+a_{j_{q-m-1}}, C_{max}\left({j^*}_{q-m-2}\right)\right\}$ and the finishing time is given by equation (52).

$$max\left\{C_1\left({j^*}_{q-m-2}\right)+a_{j_{q-m-1}}+b_{j_{q-m-1}}, C_{max}\left({j^*}_{q-m-2}\right)+b_{j_{q-m-1}}\right\} \quad (52)$$

Since there are m jobs $j_{q-m}, \dots, j_{q-3}, j_{q-2}, j_{q-1}$, by induction assumption, the finishing time of the job ${j^*}_q$ on machine B is independent of the order of jobs $j_{q-m}, \dots, j_{q-3}, j_{q-2}, j_{q-1}$ and is given by equation (53).

$$max \left\{ C_1\left({j^*}_{q-m-1}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m} a_{j_{q-i}}, C_{max}\left({j^*}_{q-m-1}\right) + b_{{j^*}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \right\} \tag{53}$$

Equation (53) is rewritten as equation (54) to save space, and substituting equation (52) in equation (54), we get equation (55).

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left({j^*}_{q-m-1}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m} a_{j_{q-i}} \\ C_{max}\left({j^*}_{q-m-1}\right) + b_{{j^*}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \end{Bmatrix} \tag{54}$$

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left(j_{q-m-1}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m} a_{j_{q-i}} \\ max \begin{Bmatrix} C_1\left({j^*}_{q-m-2}\right) + a_{j_{q-m-1}} + b_{j_{q-m-1}} \\ C_{max}\left({j^*}_{q-m-2}\right) + b_{j_{q-m-1}} \end{Bmatrix} + b_{{j^*}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \end{Bmatrix} \tag{55}$$

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left(j_{q-m-1}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m} a_{j_{q-i}} \\ C_1\left({j^*}_{q-m-2}\right) + a_{j_{q-m-1}} + b_{j_{q-m-1}} + b_{{j^*}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \\ C_{max}\left({j^*}_{q-m-2}\right) + b_{j_{q-m-1}} + b_{{j^*}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \end{Bmatrix} \tag{56}$$

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left({j^*}_{q-m-2}\right) + a_{j_{q-m-1}} + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m} a_{j_{q-i}} \\ C_1\left({j^*}_{q-m-2}\right) + a_{j_{q-m-1}} + b_{j_{q-m-1}} + b_{{j^*}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \\ C_{max}\left({j^*}_{q-m-2}\right) + b_{j_{q-m-1}} + b_{{j^*}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \end{Bmatrix} \tag{57}$$

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left({j^*}_{q-m-2}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m+1} a_{j_{q-i}} \\ C_1\left({j^*}_{q-m-2}\right) + a_{j_{q-m-1}} + b_{{j^*}_q} + \sum_{i=1}^{m+1} b_{j_{q-i}} \\ C_{max}\left({j^*}_{q-m-2}\right) + b_{{j^*}_q} + \sum_{i=1}^{m+1} b_{j_{q-i}} \end{Bmatrix} \tag{58}$$

But since $b_{j_{q-i}} < a_{j_{q-i}} \leq a_{{j^*}_q}$, the second number in (58) is less than the first number.

$$C_1\left({j^*}_{q-m-2}\right) + a_{j_{q-m-1}} + b_{{j^*}_q} + \sum_{i=1}^{m+1} b_{\mathrm{j}_{q-i}} \tag{59}$$

$$< C_1\left({j^*}_{q-m-2}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m+1} a_{\mathrm{j}_{q-i}} \tag{60}$$

The maximum of the three numbers in equation (58) reduces to a maximum of two numbers given in equation (61).

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left({j^*}_{q-m-2}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m+1} a_{j_{q-i}} \\ C_{max}\left({j^*}_{q-m-2}\right) + b_{{j^*}_q} + \sum_{i=1}^{m+1} b_{j_{q-i}} \end{Bmatrix} \tag{61}$$

If we substitute $m + 1$ in place of m in the formula in equation (34) also, we get

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left({j^*}_{q-m-1}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m} a_{j_{q-i}} \\ C_{max}\left({j^*}_{q-m-1}\right) + b_{{j^*}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \end{Bmatrix} \tag{62}$$

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left({j^*}_{q-m-1-1}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m+1} a_{j_{q-i}} \\ C_{max}\left({j^*}_{q-m-1-1}\right) + b_{{j^*}_q} + \sum_{i=1}^{m+1} b_{j_{q-i}} \end{Bmatrix} \tag{63}$$

$$C_{max}\left({j^*}_q\right) = max \begin{Bmatrix} C_1\left({j^*}_{q-m-2}\right) + a_{{j^*}_q} + b_{{j^*}_q} + \sum_{i=1}^{m+1} a_{j_{q-i}} \\ C_{max}\left({j^*}_{q-m-2}\right) + b_{{j^*}_q} + \sum_{i=1}^{m+1} b_{j_{q-i}} \end{Bmatrix} \tag{64}$$

Equation (64) is identical to equation (61).

Hence we have shown that the theorem also holds for $m+1$. Thus, by the principle of mathematical induction the theorem holds for all m such that $n_1+1 \le q-m \le q-1$.

If $q-m=n_1+1$ i. e. $m=q-n_1-1$ the formula (64) yields equations (65)-(66).

$$C_{max}\left(j^*{}_q\right) = max\begin{Bmatrix} C_1\left(j^*{}_{q-m-1}\right) + a_{j^*{}_q} + b_{j^*{}_q} + \sum_{i=1}^{m} a_{j_{q-i}} \\ C_{max}\left(j^*{}_{q-m-1}\right) + b_{j^*{}_q} + \sum_{i=1}^{m} b_{j_{q-i}} \end{Bmatrix} \tag{65}$$

$$C_{max}\left(j^*{}_q\right) = max\begin{Bmatrix} C_1\left(j^*{}_{q-(q-n_1-1)-1}\right) + a_{j^*{}_q} + b_{j^*{}_q} + \sum_{i=1}^{q-n_1-1} a_{j_{q-i}} \\ C_{max}\left(j^*{}_{q-(q-n_1-1)-1}\right) + b_{j^*{}_q} + \sum_{i=1}^{q-n_1-1} b_{j_{q-i}} \end{Bmatrix} \tag{66}$$

By simplification, the formula (66) for $C_{max}\left(j^*{}_q\right)$ reduces to the formula (67)

$$C_{max}\left(j^*{}_q\right) = max\begin{Bmatrix} C_1\left(j^*{}_{n_1}\right) + a_{j^*{}_q} + b_{j^*{}_q} + \sum_{i=1}^{q-n_1-1} a_{j_{q-i}} \\ C_{max}\left(j^*{}_{n_1}\right) + b_{j^*{}_q} + \sum_{i=1}^{q-n_1-1} b_{j_{q-i}} \end{Bmatrix} \tag{67}$$

Changing the index $q-i$ by the index n_1+i, the formula (67) for $C_{max}\left(j^*{}_q\right)$ reduces to the formula (68) and (69).

$$C_{max}\left(j^*{}_q\right) = max\begin{Bmatrix} C_1\left(j^*{}_{n_1}\right) + a_{j^*{}_q} + b_{j^*{}_q} + \sum_{i=n_1+1}^{q-1} a_{j_i} \\ C_{max}\left(j^*{}_{n_1}\right) + b_{j^*{}_q} + \sum_{i=n_1+1}^{q-1} b_{j_i} \end{Bmatrix} \tag{68}$$

$$C_{max}\left(j^*{}_q\right) = max\begin{Bmatrix} C_1\left(j^*{}_{n_1}\right) + b_{j^*{}_q} + \sum_{i=n_1+1}^{q} a_{j_i} \\ C_{max}\left(j^*{}_{n_1}\right) + \sum_{i=n_1+1}^{q} b_{j_i} \end{Bmatrix} \tag{69}$$

Formula (69) gives the completion time of the job $j^*{}_q$ explicitly as maximum of the sum of processing times of the jobs in the two machines and that of job $j^*{}_q$. In this formula the maximum of the two numbers $C_1\left(j^*{}_{n_1}\right) + b_{j^*{}_q} + \sum_{i=n_1+1}^{q} a_{j_i}$ and $C_{max}\left(j^*{}_{n_1}\right) + \sum_{i=n_1+1}^{q} b_{j_i}$ is symmetric with respect to jobs and so it is independent of the order of operations. Thus, the theorem was proved.

Since the jobs $j_{n_1+1}, \dots, j_{q-2}, j_{q-1}$ are permutations of the jobs $j^*{}_{n_1+1}, \dots, j^*{}_{q-2}, j^*{}_{q-1}$, and the optimal sequence is also obtained by one of these permutations and the total elapsed time in the optimal sequence to finish q jobs $C_{max}\left(j^*{}_q\right)$given by formula (69) reduces to formula (70).

$$C_{max}\left(j^*{}_q\right) = max\begin{Bmatrix} C_1\left(j^*{}_{n_1}\right) + b_{j^*{}_q} + \sum_{i=n_1+1}^{q} a_{j^*{}_i} \\ C_{max}\left(j^*{}_{n_1}\right) + \sum_{i=n_1+1}^{q} b_{j^*{}_i} \end{Bmatrix} \tag{70}$$

If further we keep all jobs in J_1 fixed in the optimal sequence, the total elapsed time to finish all jobs with the above freedom will be equal to the optimal value. Hence any sequence obtained by the above sequencing freedom is optimal. Since we are free to arrange the jobs $j^*{}_{n_1+1} \to \cdots \to j^*{}_{q-2} \to j^*{}_{q-1}$ in an arbitrary order, the above machine sequencing freedom creates $(q-n_1-1)!$ sequences, all of which are optimal. For reference, we call the job $j^*{}_q$ in the above formulation the maximal job of $\left\{j^*{}_i; i = 1, 2, 3, \dots, n\right\} = j^*{}_1 \to j^*{}_2 \to j^*{}_3 \to \cdots \to j^*{}_n$ and we call the block of jobs $j^*{}_{n_1+1} \to \dots \to j^*{}_{q-3} \to j^*{}_{q-2} \to j^*{}_{q-1}$ the free jobs of second kind.

Concluding Remark

Combining the results of Section 4.2.1 and Section 4.2.2, by substituting $C_{max}\left({j^*}_{n_1}\right)$ in formula (70) by the right hand side in formula (33), $C_{max}\left({j^*}_{q}\right)$ reduces to the formula (71) and (72).

$$C_{max}\left({j^*}_{q}\right) = max\begin{Bmatrix} C_1\left({j^*}_k\right) + \sum_{i=k+1}^{n_1} a_{{j^*}_i} + b_{{j^*}_q} + \sum_{i=n_1+1}^{q} a_{{j^*}_i} \\ C_{max}\left({j^*}_k\right) + \sum_{i=k+1}^{n_1} b_{{j^*}_i} + \sum_{i=n_1+1}^{q} b_{{j^*}_i} \end{Bmatrix} \tag{71}$$

$$C_{max}\left({j^*}_{q}\right) = max\begin{Bmatrix} C_1\left({j^*}_k\right) + b_{{j^*}_q} + \sum_{i=k+1}^{q} a_{{j^*}_i} \\ C_{max}\left({j^*}_k\right) + \sum_{i=k+1}^{q} b_{{j^*}_i} \end{Bmatrix} \tag{72}$$

At this point by looking at the formula in (72) only, we suggest not to conclude that the union of the free jobs of first and second kind could also be arranged in an arbitrary order because there are just a few particular cases for which this conclusion does not hold true. Thus, we need to compute the sequence dependent starting and completion times of the remaining jobs ${j^*}_{q+1} \to {j^*}_{q+2} \to {j^*}_{q+3} \to \cdots \to {j^*}_n$ on the two machines to find total elapsed time.

4.3. Algorithm: Dimension Reduction

Thus we have proved the following algorithm.
Let problem $P = \{j_i = (a_{j_i}, b_{j_i});\ i = 1, 2, 3, \ldots, n\}$ with no passing of jobs on the two machines in the order $A \to B$ be given where a_{j_i} and b_{j_i} are processing times of job j_i on machines A and B, respectively for $i = 1, 2, 3, \ldots, n$.

- *Step 1:*

Partition all jobs in P into two clusters, J_1 and J_2 where $J_1 = \left\{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} \le b_{j_i};\ i = 1, 2, 3, \ldots, n\right\}$ and $J_2 = \left\{j_i = (a_{j_i}, b_{j_i}) \in P: a_{j_i} > b_{j_i};\ i = 1, 2, 3, \ldots, n\right\}$. Identify all jobs in J_1 and J_2. Let there be n_1 jobs in J_1 and n_2 jobs in J_2.

- *Step 2:*

Let $\bar{A} = \left\{a_{j_i}: (a_{j_i}, b_{j_i}) \in J_1\right\}$ be the set of distinct operation times on machine A for all jobs of the first kind and let $\bar{B} = \left\{b_{j_i}: (a_{j_i}, b_{j_i}) \in J_2\right\}$ be the set of distinct operation times on machine B for all jobs of the second kind.

- *Step 3:*

Define $f_1: \bar{A} \to \Re$ by the map given by $f_1\left(a_{j_l}\right) = max_{\left\{a_{j_i} = a_{j_l}\right\}}\left\{b_{j_i}: (a_{j_i}, b_{j_i}) \in J_1\right\}$ $\quad for\ all\ a_{j_l} \in \bar{A}$. This definition assigns the highest processing time on machine B for each distinct processing time on machine A.

- *Step 4:*

Define $f_2: \bar{B} \to \Re$ by the map given by $f_2\left(b_{j_l}\right) = max_{\left\{b_{j_i} = b_{j_l}\right\}}\left\{a_{j_i}: (a_{j_i}, b_{j_i}) \in J_2\right\}$ $\quad , for\ all\ b_{j_l} \in \bar{B}$. This definition assigns the highest processing time on machine A for each distinct processing time on machine B.

$\overline{J_1} = \left\{\left(a_{j_i}, f_1\left(a_{j_i}\right)\right)\right\} for\ all\ a_{j_i} \in \bar{A}$ and $\overline{J_2} = \left\{\left(f_2\left(b_{j_l}\right), b_{j_i}\right)\right\} for\ all\ b_{j_i} \in \bar{B}$ give distinct equivalence classes.

- *Step 5:*

Therefore, the reduced problem at the end of the first phase of dimension reduction becomes sequencing jobs in $\bar{P} = \overline{J_1} \cup \overline{J_2}$.

- *Step 6:*

Let $b_1 = max_{\left\{a_{j_i} \in \bar{A}\right\}}\left\{f_1\left(a_{j_i}\right)\right\}$ and $a_1 = max_{\left\{b_{j_i} \in \bar{B}\right\}}\left\{f_2\left(b_{j_i}\right)\right\}$ Let $a^* = min_{\left\{f_1\left(a_{j_i}\right) = b_1\right\}}\left\{a_{j_i}\right\}$ and $b^* = min_{\left\{f_2\left(b_{j_i}\right) = a_1\right\}}\left\{b_{j_i}\right\}$

- *Step 7a:*

Identify the job $j_{l_1} = (a_{j_{l_1}}, b_{j_{l_1}}) \in J_1$ such that $a_{j_{l_1}} + b_{j_{l_1}} = max_{a_{j_i} > a^*}\left\{a_{j_i} + b_{j_i}, (a_{j_i}, b_{j_i}) \in J_1\right\}$.Identify the job $j_{l_2} = (a_{j_{l_2}}, b_{j_{l_2}}) \in J_2$ such that $a_{j_{l_2}} + b_{j_{l_2}} = max_{b_{j_i} > b^*}\left\{a_{j_i} + b_{j_i}, (a_{j_i}, b_{j_i}) \in J_2\right\}$

- *Step 7b:*

$\overline{\overline{J_1}} = \left\{\left(a_{j_i}, f_1\left(a_{j_l}\right)\right): a_{j_i} \le a^* for\ all\ a_{j_i} \in \bar{A}\right\} \cup \left\{j_{l_1}\right\}$ and $\overline{\overline{J_2}} = \left\{\left(f_2\left(b_{j_l}\right), b_{j_i}\right) for\ all\ b_{j_i} \in \bar{B}\right\} \cup \left\{j_{l_2}\right\}$.

- *Step 8*

Therefore, the reduced problem at the end of the second phase of dimension reduction becomes sequencing jobs in $\overline{\overline{P}} = \overline{\overline{J_1}} \cup \overline{\overline{J_2}}$.

4.4. Algorithm: Relaxation of Johnson's Algorithm

The Johnson's algorithm is an exact solution method of the two machines, one-way, no-passing scheduling tasks problem, which serves as a basis for many heuristic algorithms. This rule is a complete list of ordering the jobs by filling the first or the last available space based on minimum operation times in the two machines from the waiting list until, finally, only one free space and one last job to be assigned remain in the waiting list. We make $n!$ comparisons to obtain the optimal sequence. To overcome the problem of computation time, the current study identified a relaxation of Johnson's algorithm by developing an early stopover criteria due to the fact that after listing only some critical jobs at the beginning and end of the optimal sequence using Johnson's procedure, it does not matter in whichever order the remaining jobs are operated as far as makespan is concerned.

- *Step 0:*

Compute $d = b_{j_i} - a_{j_i};\ i = 1, 2, 3, \dots, n.$ Waiting list 1 is $J_1 = \left\{j_i = \left(a_{j_i}, b_{j_i}\right) \in P: b_{j_i} - a_{j_i} \geq 0;\ i = 1, 2, 3, \dots, n\right\}$. Waiting list 2 is $J_2 = \left\{j_i = \left(a_{j_i}, b_{j_i}\right) \in P: b_{j_i} - a_{j_i} < 0;\ i = 1, 2, 3, \dots, n\right\}$.

- *Step 1a*

Identify the job $j^*{}_k = (a_{j^*{}_k}, b_{j^*{}_k})$ such that $b_{j^*{}_k} = max_{(a_{j_i}, b_{j_i}) \in J_1}\left\{b_{j_i}\right\}$ and $a_{j^*{}_k} = min_{b_{j_i} = b_{j^*{}_k}}\left\{a_{j_i}, (a_{j_i}, b_{j_i}) \in J_1\right\}$

- *Step 1b*

Identify the job $j^*{}_q = (a_{j^*{}_q}, b_{j^*{}_q})$ such that $a_{j^*{}_q} = max_{(a_{j_i}, b_{j_i}) \in J_2}\left\{a_{j_i}\right\}$ and $b_{j^*{}_q} = min_{a_{j_i} = a_{j^*{}_q}}\left\{b_{j_i}, (a_{j_i}, b_{j_i}) \in J_2\right\}$

- *Step 2 (Johnson's algorithm main)*

Examine the columns of A and B for processing times on machines A and B and find the smallest processing time among unscheduled jobs (waiting list). Apply Johnson's algorithm on P and remove the scheduled job from P, waiting list 1 and waiting list 2 until jobs $j^*{}_k$ or $j^*{}_q$ are assigned. Go to step 3.

- *Step 3a (Mini-max criteria 1)*

If $j^*{}_k$ is scheduled, then remove all unscheduled jobs in waiting list 1 from P and terminate this step. Go to step 2.

- *Step 3b (Mini-max criteria 2)*

If $j^*{}_q$ is scheduled, then remove all unscheduled jobs in waiting list 2 from P and terminate this step. Go to step 2.

- *Step 4 (Relaxation of Johnson's algorithm)*

If both $j^*{}_k$ and $j^*{}_q$ are assigned, then terminate Johnson's algorithm main. Go to step 5.

- *Step 5a*

If waiting list 1 is non-empty, then choose at random a job in waiting list 1 and assign the corresponding job in the first available position in sequence and remove the assigned job from the waiting list 1. Repeat this step until waiting list 1 is empty.

- *Step 5b*

If waiting list 2 is non-empty then choose at random a job in waiting list 2 and assign the corresponding job in the last available position in sequence and remove the assigned job from the waiting list 2. Repeat this step until waiting list 2 is empty.

- *Step 6*

Repeat steps 5a and 5b until all P, waiting list 1 and waiting list 2 are empty.
In the relaxation algorithm above, we did not violate Johnson's algorithm except termination criteria. Thus, ties for jobs with equal processing time on the two machines may be broken arbitrarily.

5. Illustrative Example

In the following example, The Two Machines Flow Shop Sequencing Problem of 100 jobs indexed $j_1, j_2, j_3 \cdots, j_{100}$ was generated from a normal distribution $N[\mu, \sigma^2]$ with mean μ and standard deviation σ^2 in Microsoft Excel spreadsheet user interface for different values of the parameters. The processing time on machine A was assumed to follow a normal distribution with mean $\mu = 58$ and standard deviation $\sigma^2 = 2$ and the processing time on machine B was assumed to follow a normal distribution with mean $\mu = 51$ and standard deviation $\sigma^2 = 8$. The procedure was as follows: in columns 2 and 3 random numbers were generated from $[0, 1]$ representing the probabilities of processing times on machines A and B respectively. In column 4 we computed the inverse of the Cumulative Normal Distribution Function for probability values defined in column 2, with distribution mean $\mu = 58$ and standard deviation $\sigma^2 = 2$. Similarly, in column 5 we computed the inverse of the Cumulative Normal Distribution Function for probability values defined in column 3, with distribution mean $\mu = 51$ and standard deviation $\sigma^2 = 8$. Thus, the numbers in columns 4 and 5 represent the corresponding processing times on machines A and B. In column 6, jobs were assigned the value 0 if the value in column 4 was less than or equal to the value in column 5, and 1 otherwise to partition all jobs into two clusters J_1 and J_2 as explained before. Then the jobs were sorted in ascending value on column 6 to arrange all jobs of the first kind before jobs of the second kind. Next all jobs of the first kind only were selected and sorted by ascending value on column 4 to arrange all jobs of the first kind in a non-decreasing order of processing time on the first machine. Similarly, all jobs of the second kind only were selected and sorted in ascending value on column 5 to arrange all jobs of the second kind in a non-increasing order of processing time on the second machine. The resulting sequence was, therefore, an optimal Johnson's sequence and the corresponding sequence positions for all jobs were assigned in column 1, starting from beginning to end. Table 1 below gives the results.

The formulas for starting and completion of jobs on the first machine A were entered in columns 7 and 8. Similarly, the formulas for starting and completion of jobs on second machine B were entered in columns 9 and 10. The formula to calculate the idle time (slack time) due to each job was entered in column 11 so that the formulas automatically run for any other permutations of the jobs.

After identifying the optimal Johnson's sequence, applying the mini-max criteria discussed earlier, the maximum processing time on the second machine for all jobs of the first kind was identified to be 65 , and its first occurrence was in the j^*_8 in the optimal Johnson's sequence. Similarly applying the mini-max criteria discussed earlier, the maximum processing time on the first machine for all jobs of the second kind was identified to be 63 and its last occurrence was in the j^*_{82} in the optimal Johnson's sequence.

In the next step the concept of random numbers was used to generate multiple alternate optimal sequences and to verify the findings of this study. In column 12, a new sequence in terms of random numbers was defined for all jobs as follows. For the jobs up to and including the minimal job, the same sequence order as the optimal Johnson's sequence was maintained. Also, for jobs starting from the maximal job onward, the same sequence order as the optimal Johnson's sequence was maintained. But for free jobs of the first kind, a random number between 0 and 1 was added to the minimal job position number, and for free jobs of the second kind a random number between 0 and 1 was subtracted from the maximal job position and all jobs are sorted in increasing order of the values in column 12. Then a new sequence was defined in column 13 to get an alternate optimal sequence. Then comparisons were made between the Johnson's sequence in Table 1 and the alternate sequence in Table 2.

Table 1. Optimal Johnson's sequence for the generated problem in the illustration example.

Joh seq	Rand A	Rand B	A	B	J1/ J2	In A	Out A	In B	Out B	Idle	Relax	Alt seq
1	0.071	0.782	55	57	0	0	55	55	112	55	1.000	1
2	0.060	0.837	55	59	0	55	110	112	171	0	2.000	2
3	0.070	0.718	55	56	0	110	165	171	227	0	3.000	3
4	0.163	0.786	56	57	0	165	221	227	284	0	4.000	4
5	0.106	0.743	56	56	0	221	277	284	340	0	5.000	5
6	0.157	0.951	56	64	0	277	333	340	404	0	6.000	6
7	0.211	0.951	56	64	0	333	389	404	468	0	7.000	7
8	0.327	0.954	57	**65**	0	389	**446**	468	**533**	0	8.000	8
9	0.676	0.972	57	62	0	446	503	533	595	0	9.173	12
10	0.694	0.852	57	60	0	503	560	595	655	0	9.398	15
11	0.630	0.834	58	59	0	560	618	655	714	0	9.931	22
12	0.323	0.915	58	64	0	618	676	714	778	0	9.401	16
13	0.693	0.864	58	62	0	676	734	778	840	0	9.480	18
14	0.805	0.870	59	60	0	734	793	840	900	0	9.306	13
15	0.379	0.862	59	61	0	793	852	900	961	0	9.505	19
16	0.572	0.947	59	63	0	852	911	961	1024	0	9.995	23
17	0.852	0.995	59	61	0	911	970	1024	1085	0	9.016	9
18	0.503	0.909	59	59	0	970	1029	1085	1144	0	9.046	10
19	0.718	0.897	59	59	0	1029	1088	1144	1203	0	9.108	11
20	0.964	0.990	60	62	0	1088	1148	1203	1265	0	9.475	17
21	0.922	0.920	60	60	0	1148	1208	1265	1325	0	9.322	14
22	0.543	0.833	61	62	0	1208	1269	1325	1387	0	9.843	21
23	0.683	0.934	62	64	0	1269	**1331**	1387	**1451**	0	9.626	20
24	0.770	0.675	58	57	1	1331	1389	1451	1508	0	24.300	43
25	0.785	0.715	61	57	1	1389	1450	1508	1565	0	24.570	58
26	0.376	0.415	60	56	1	1450	1510	1565	1621	0	24.046	25
27	0.029	0.555	57	56	1	1510	1567	1621	1677	0	24.870	72
28	0.274	0.310	57	56	1	1567	1624	1677	1733	0	24.199	37

29	0.578	0.497	60	56	1	1624	1684	1733	1789	0	24.434	53
30	0.755	0.500	57	55	1	1684	1741	1789	1844	0	24.443	54
31	0.943	0.323	59	55	1	1741	1800	1844	1899	0	24.013	24
32	0.855	0.396	56	55	1	1800	1856	1899	1954	0	24.924	77
33	0.970	0.458	55	54	1	1856	1911	1954	2008	0	24.987	80
34	0.143	0.582	55	54	1	1911	1966	2008	2062	0	24.663	66
35	0.320	0.269	57	54	1	1966	2023	2062	2116	0	24.654	64
36	0.909	0.432	61	54	1	2023	2084	2116	2170	0	24.384	51
37	0.270	0.731	56	53	1	2084	2140	2170	2223	0	24.185	34
38	0.727	0.561	60	53	1	2140	2200	2223	2276	0	24.583	61
39	0.456	0.534	56	53	1	2200	2256	2276	2329	0	24.357	47
40	0.766	0.567	59	53	1	2256	2315	2329	2382	0	24.848	71
41	0.316	0.277	58	53	1	2315	2373	2382	2435	0	24.489	55
42	0.855	0.225	59	52	1	2373	2432	2435	2487	0	24.200	38
43	0.461	0.768	54	52	1	2432	2486	2487	2539	0	24.050	27
44	0.785	0.254	58	52	1	2486	2544	2544	2596	5	24.202	39
45	0.946	0.380	59	52	1	2544	2603	2603	2655	7	24.240	40
46	0.104	0.389	58	51	1	2603	2661	2661	2712	6	24.372	48
47	0.178	0.608	60	51	1	2661	2721	2721	2772	9	24.616	62
48	0.493	0.477	57	51	1	2721	2778	2778	2829	6	24.891	74
49	0.235	0.217	58	51	1	2778	2836	2836	2887	7	24.084	29
50	0.617	0.354	60	51	1	2836	2896	2896	2947	9	24.535	57
51	0.951	0.645	58	51	1	2896	2954	2954	3005	7	24.907	75
52	0.370	0.230	59	51	1	2954	3013	3013	3064	8	24.087	30
53	0.806	0.734	61	50	1	3013	3074	3074	3124	10	24.196	36
54	0.229	0.705	61	50	1	3074	3135	3135	3185	11	24.935	79
55	0.463	0.591	62	50	1	3135	3197	3197	3247	12	24.158	33
56	0.663	0.397	58	50	1	3197	3255	3255	3305	8	24.919	76
57	0.839	0.498	57	50	1	3255	3312	3312	3362	7	24.742	67
58	0.938	0.768	58	50	1	3312	3370	3370	3420	8	24.837	70
59	0.464	0.279	61	49	1	3370	3431	3431	3480	11	24.316	45
60	0.395	0.404	55	49	1	3431	3486	3486	3535	6	24.347	46
61	0.827	0.591	57	49	1	3486	3543	3543	3592	8	24.047	26
62	0.806	0.499	59	49	1	3543	3602	3602	3651	10	24.526	56
63	0.394	0.359	57	49	1	3602	3659	3659	3708	8	24.581	60
64	0.280	0.637	60	49	1	3659	3719	3719	3768	11	24.137	32
65	0.436	0.178	57	48	1	3719	3776	3776	3824	8	24.781	68
66	0.096	0.643	59	48	1	3776	3835	3835	3883	11	24.383	50
67	0.268	0.459	57	48	1	3835	3892	3892	3940	9	24.647	63
68	0.382	0.376	57	47	1	3892	3949	3949	3996	9	24.056	28
69	0.484	0.250	61	47	1	3949	4010	4010	4057	14	24.122	31
70	0.423	0.437	63	47	1	4010	4073	4073	4120	16	24.926	78
71	0.599	0.595	58	46	1	4073	4131	4131	4177	11	24.798	69
72	0.305	0.735	58	46	1	4131	4189	4189	4235	12	24.572	59
73	0.023	0.265	57	46	1	4189	4246	4246	4292	11	24.267	41
74	0.377	0.505	54	46	1	4246	4300	4300	4346	8	24.881	73
75	0.454	0.512	60	46	1	4300	4360	4360	4406	14	24.306	44
76	0.549	0.446	57	46	1	4360	4417	4417	4463	11	24.188	35
77	0.164	0.711	57	45	1	4417	4474	4474	4519	11	24.376	49
78	0.996	0.297	57	45	1	4474	4531	4531	4576	12	24.999	81
79	0.951	0.460	57	45	1	4531	4588	4588	4633	12	24.397	52
80	0.067	0.652	60	45	1	4588	4648	4648	4693	15	24.284	42
81	0.235	0.226	58	44	1	4648	4706	4706	4750	13	24.662	65
82	0.990	0.189	**63**	44	1	4706	**4769**	4769	**4813**	19	82.000	82
83	0.220	0.172	56	43	1	4769	4825	4825	4868	12	83.000	83
84	0.475	0.147	58	43	1	4825	4883	4883	4926	15	84.000	84
85	0.465	0.154	58	43	1	4883	4941	4941	4984	15	85.000	85
86	0.757	0.152	59	43	1	4941	5000	5000	5043	16	86.000	86

87	0.855	0.157	60	43	1	5000	5060	5060	5103	17	87.000	87
88	0.411	0.126	58	42	1	5060	5118	5118	5160	15	88.000	88
89	0.482	0.108	58	41	1	5118	5176	5176	5217	16	89.000	89
90	0.548	0.101	58	41	1	5176	5234	5234	5275	17	90.000	90
91	0.623	0.086	59	40	1	5234	5293	5293	5333	18	91.000	91
92	0.626	0.083	59	40	1	5293	5352	5352	5392	19	92.000	92
93	0.758	0.070	59	39	1	5352	5411	5411	5450	19	93.000	93
94	0.812	0.074	60	39	1	5411	5471	5471	5510	21	94.000	94
95	0.575	0.059	58	38	1	5471	5529	5529	5567	19	95.000	95
96	0.279	0.051	57	38	1	5529	5586	5586	5624	19	96.000	96
97	0.159	0.040	56	37	1	5586	5642	5642	5679	18	97.000	97
98	0.266	0.038	57	37	1	5642	5699	5699	5736	20	98.000	98
99	0.853	0.030	60	36	1	5699	5759	5759	5795	23	99.000	99
100	0.744	0.017	59	34	1	5759	**5818**	5818	**5852**	23	100.000	100
										767		

Observe first from Table 1 the following values:
Machine A completes the minimal job at time $C_1\left(j^*{}_8\right) = \mathbf{446}$ and machine B completes it at time $C_{max}\left(j^*{}_8\right) = \mathbf{533}$. Machine A completes all jobs of the first kind at time $C_1\left(j^*{}_{23}\right) = \mathbf{1331}$ and machine B completes the same jobs at time $C_{max}\left(j^*{}_{23}\right) = \mathbf{1451}$. Machine A completes the maximal job at time $C_1\left(j^*{}_{82}\right) = \mathbf{4769}$ and machine B completes it at time $C_{max}\left(j^*{}_{82}\right) = \mathbf{4813}$. Machine A completes the last job at time $C_1\left(j^*{}_{100}\right) = \mathbf{5818}$ and machine B completes it at $C_{max}\left(j^*{}_{100}\right) = \mathbf{5852}$. All the values described here are outlined in Table 1 by dark line in the corresponding rows containing the values.
Also observe the following values:
The sum of the processing times of free jobs of the first kind on machine B is $\sum_{i=9}^{23} b_{j^*{}_i} = \mathbf{918}$. Thus, $C_{max}\left(j^*{}_8\right) + \sum_{i=9}^{23} b_{j^*{}_i} = 533 + 918 = \mathbf{1451} = C_{max}\left(j^*{}_{23}\right)$. Thus equation (6) is verified.
The sum of the processing times of free jobs of the second kind on machine A is $\sum_{i=24}^{(81)} a_{j^*{}_i} = \mathbf{3375}$ and the sum of the processing times of free jobs of the second kind on machine B is $\sum_{i=24}^{(81)} a_{j^*{}_i} = \mathbf{2928}$.
Therefore, $C_1\left(j^*{}_{23}\right) + a_{j^*{}_{82}} + b_{j^*{}_{82}} + \sum_{i=24}^{(81)} a_{j^*{}_i} = 1331 + 63 + 44 + 3375 = \mathbf{4813}$ and $C_{max}\left(j^*{}_{23}\right) + b_{j^*{}_{82}} + \sum_{i=24}^{81} b_{j^*{}_i} = 1451 + 44 + 2928 = \mathbf{4423}$.
Thus, $max\left\{C_1\left(j^*{}_{23}\right) + a_{j^*{}_{82}} + b_{j^*{}_{82}} + \sum_{i=24}^{(81)} a_{j^*{}_i}, C_{max}\left(j^*{}_{23}\right) + b_{j^*{}_{82}} + \sum_{i=24}^{81} b_{j^*{}_i}\right\}$
$= max\{4813, 4423\} = \mathbf{4813} = C_{max}\left(j^*{}_{82}\right)$. Thus equation (71) is verified.
The total elapsed time is $C_{max}\left(j^*{}_{100}\right) = \mathbf{5852}$

The total idle time for machine B was $\mathrm{I} = 767$ (see the value at the end of column 11 in Tables 1 and 2).
It remains to verify Theorems 1 and 2. We show this by sorting the jobs in optimal Johnson's sequence in ascending order of the values assigned to the jobs in column 12, and compare the finishing times of jobs of the first kind as well as the maximal job. To do this, we copy all the values in the **bold** cells for Johnson's sequence (Table 1) and sort the jobs in ascending value in column 12. This gives a different sequence of jobs.

Table 2. Alternate optimal sequence for the generated problem in the illustration example.

Joh seq	Rand A	Rand B	A	B	J1/ J2	In A	Out A	In B	Out B	Idle	Relax	Alt seq
1	0.071	0.782	55	57	0	0	55	55	112	55	1.000	1
2	0.060	0.837	55	59	0	55	110	112	171	0	2.000	2
3	0.070	0.718	55	56	0	110	165	171	227	0	3.000	3
4	0.163	0.786	56	57	0	165	221	227	284	0	4.000	4
5	0.106	0.743	56	56	0	221	277	284	340	0	5.000	5
6	0.157	0.951	56	64	0	277	333	340	404	0	6.000	6
7	0.211	0.951	56	64	0	333	389	404	468	0	7.000	7
8	0.327	0.954	57	**65**	0	389	**446**	468	**533**	0	8.000	8
17	0.852	0.995	59	61	0	446	505	533	594	0	9.173	9

18	0.503	0.909	59	59	0	505	564	594	653	0	9.398	10
19	0.718	0.897	59	59	0	564	623	653	712	0	9.931	11
9	0.676	0.972	57	62	0	623	680	712	774	0	9.401	12
14	0.805	0.870	59	60	0	680	739	774	834	0	9.480	13
21	0.922	0.920	60	60	0	739	799	834	894	0	9.306	14
10	0.694	0.852	57	60	0	799	856	894	954	0	9.505	15
12	0.323	0.915	58	64	0	856	914	954	1018	0	9.995	16
20	0.964	0.990	60	62	0	914	974	1018	1080	0	9.016	17
13	0.693	0.864	58	62	0	974	1032	1080	1142	0	9.046	18
15	0.379	0.862	59	61	0	1032	1091	1142	1203	0	9.108	19
23	0.683	0.934	62	64	0	1091	1153	1203	1267	0	9.475	20
22	0.543	0.833	61	62	0	1153	1214	1267	1329	0	9.322	21
11	0.630	0.834	58	59	0	1214	1272	1329	1388	0	9.843	22
16	0.572	0.947	59	63	0	1272	**1331**	1388	**1451**	0	9.626	23
31	0.943	0.323	59	55	1	1331	1390	1451	1506	0	24.300	24
26	0.376	0.415	60	56	1	1390	1450	1506	1562	0	24.570	25
61	0.827	0.591	57	49	1	1450	1507	1562	1611	0	24.046	26
43	0.461	0.768	54	52	1	1507	1561	1611	1663	0	24.870	27
68	0.382	0.376	57	47	1	1561	1618	1663	1710	0	24.199	28
49	0.235	0.217	58	51	1	1618	1676	1710	1761	0	24.434	29
52	0.370	0.230	59	51	1	1676	1735	1761	1812	0	24.443	30
69	0.484	0.250	61	47	1	1735	1796	1812	1859	0	24.013	31
64	0.280	0.637	60	49	1	1796	1856	1859	1908	0	24.924	32
55	0.463	0.591	62	50	1	1856	1918	1918	1968	10	24.987	33
37	0.270	0.731	56	53	1	1918	1974	1974	2027	6	24.663	34
76	0.549	0.446	57	46	1	1974	2031	2031	2077	4	24.654	35
53	0.806	0.734	61	50	1	2031	2092	2092	2142	15	24.384	36
28	0.274	0.310	57	56	1	2092	2149	2149	2205	7	24.185	37
42	0.855	0.225	59	52	1	2149	2208	2208	2260	3	24.583	38
44	0.785	0.254	58	52	1	2208	2266	2266	2318	6	24.357	39
45	0.946	0.380	59	52	1	2266	2325	2325	2377	7	24.848	40
73	0.023	0.265	57	46	1	2325	2382	2382	2428	5	24.489	41
80	0.067	0.652	60	45	1	2382	2442	2442	2487	14	24.200	42
24	0.770	0.675	58	57	1	2442	2500	2500	2557	13	24.050	43
75	0.454	0.512	60	46	1	2500	2560	2560	2606	3	24.202	44
59	0.464	0.279	61	49	1	2560	2621	2621	2670	15	24.240	45
60	0.395	0.404	55	49	1	2621	2676	2676	2725	6	24.372	46
39	0.456	0.534	56	53	1	2676	2732	2732	2785	7	24.616	47
46	0.104	0.389	58	51	1	2732	2790	2790	2841	5	24.891	48
77	0.164	0.711	57	45	1	2790	2847	2847	2892	6	24.084	49
66	0.096	0.643	59	48	1	2847	2906	2906	2954	14	24.535	50
36	0.909	0.432	61	54	1	2906	2967	2967	3021	13	24.907	51
79	0.951	0.460	57	45	1	2967	3024	3024	3069	3	24.087	52
29	0.578	0.497	60	56	1	3024	3084	3084	3140	15	24.196	53
30	0.755	0.500	57	55	1	3084	3141	3141	3196	1	24.935	54
41	0.316	0.277	58	53	1	3141	3199	3199	3252	3	24.158	55
62	0.806	0.499	59	49	1	3199	3258	3258	3307	6	24.919	56
50	0.617	0.354	60	51	1	3258	3318	3318	3369	11	24.742	57
25	0.785	0.715	61	57	1	3318	3379	3379	3436	10	24.837	58
72	0.305	0.735	58	46	1	3379	3437	3437	3483	1	24.316	59
63	0.394	0.359	57	49	1	3437	3494	3494	3543	11	24.347	60
38	0.727	0.561	60	53	1	3494	3554	3554	3607	11	24.047	61
47	0.178	0.608	60	51	1	3554	3614	3614	3665	7	24.526	62
67	0.268	0.459	57	48	1	3614	3671	3671	3719	6	24.581	63
35	0.320	0.269	57	54	1	3671	3728	3728	3782	9	24.137	64
81	0.235	0.226	58	44	1	3728	3786	3786	3830	4	24.781	65

34	0.143	0.582	55	54	1	3786	3841	3841	3895	11	24.383	66
57	0.839	0.498	57	50	1	3841	3898	3898	3948	3	24.647	67
65	0.436	0.178	57	48	1	3898	3955	3955	4003	7	24.056	68
71	0.599	0.595	58	46	1	3955	4013	4013	4059	10	24.122	69
58	0.938	0.768	58	50	1	4013	4071	4071	4121	12	24.926	70
40	0.766	0.567	59	53	1	4071	4130	4130	4183	9	24.798	71
27	0.029	0.555	57	56	1	4130	4187	4187	4243	4	24.572	72
74	0.377	0.505	54	46	1	4187	4241	4243	4289	0	24.267	73
48	0.493	0.477	57	51	1	4241	4298	4298	4349	9	24.881	74
51	0.951	0.645	58	51	1	4298	4356	4356	4407	7	24.306	75
56	0.663	0.397	58	50	1	4356	4414	4414	4464	7	24.188	76
32	0.855	0.396	56	55	1	4414	4470	4470	4525	6	24.376	77
70	0.423	0.437	63	47	1	4470	4533	4533	4580	8	24.999	78
54	0.229	0.705	61	50	1	4533	4594	4594	4644	14	24.397	79
33	0.970	0.458	55	54	1	4594	4649	4649	4703	5	24.284	80
78	0.996	0.297	57	45	1	4649	4706	4706	4751	3	24.662	81
82	0.990	0.189	**63**	44	1	4706	**4769**	4769	**4813**	18	82.000	82
83	0.220	0.172	56	43	1	4769	4825	4825	4868	12	83.000	83
84	0.475	0.147	58	43	1	4825	4883	4883	4926	15	84.000	84
85	0.465	0.154	58	43	1	4883	4941	4941	4984	15	85.000	85
86	0.757	0.152	59	43	1	4941	5000	5000	5043	16	86.000	86
87	0.855	0.157	60	43	1	5000	5060	5060	5103	17	87.000	87
88	0.411	0.126	58	42	1	5060	5118	5118	5160	15	88.000	88
89	0.482	0.108	58	41	1	5118	5176	5176	5217	16	89.000	89
90	0.548	0.101	58	41	1	5176	5234	5234	5275	17	90.000	90
91	0.623	0.086	59	40	1	5234	5293	5293	5333	18	91.000	91
92	0.626	0.083	59	40	1	5293	5352	5352	5392	19	92.000	92
93	0.758	0.070	59	39	1	5352	5411	5411	5450	19	93.000	93
94	0.812	0.074	60	39	1	5411	5471	5471	5510	21	94.000	94
95	0.575	0.059	58	38	1	5471	5529	5529	5567	19	95.000	95
96	0.279	0.051	57	38	1	5529	5586	5586	5624	19	96.000	96
97	0.159	0.040	56	37	1	5586	5642	5642	5679	18	97.000	97
98	0.266	0.038	57	37	1	5642	5699	5699	5736	20	98.000	98
99	0.853	0.030	60	36	1	5699	5759	5759	5795	23	99.000	99
100	0.744	0.017	59	34	1	5759	**5818**	5818	**5852**	23	100.00	100
										767		

Table 2 was obtained from Table 1 by sorting in ascending values on column 12.

To check that it is also the optimal sequence, we checked the values in the **bold** cells with those recorded for Johnson's sequence (Table 1); in particular we checked the total elapsed time $C_{max}\left(j^*_{100}\right) = \mathbf{5852}$ for each "sort ascending values on column 12" instruction. It can easily be verified that Table 2 also gives an optimal sequence irrespective of the order of free jobs of the first kind and free jobs of the second kind if they are scheduled together. Thus, the results of the two theorems were verified with this example.

Our final conclusion about dimension reduction was made in reference to the above example. At termination of Johnson's algorithm, the problem size was reduced to those jobs that hold fixed position in all the alternate optimal sequences. In the above example, the problem size was reduced from 100 to only 29 jobs (8 jobs at the beginning, 19 jobs at the end and a representative job each for free jobs of the two kinds. This is a significant dimension reduction at very low cost of computation. This may be expressed as a 71% decrease in problem size. There are at least $15! * 57!$ alternate optimal sequences for the illustrative example above obtained by this procedure only.

6. Summary, Conclusion and Recommendation

The purpose of this study was to apply dimension reduction methods for Flow Shop Scheduling Problems to decrease problem size. The development of solution methods for the 'n -Jobs m –Machines' Flow Shop Scheduling Problems was limited by the number of jobs n and number of machines m. Due to these difficulties solutions have been developed either for a small number of jobs or a small number of machines. To enable solution methods to be applicable for a large number of jobs, it was important to cluster jobs into principal components by defining a projective mapping.

This removes the redundant information in the original problem. Then solution methods are needed for only targeted jobs and once the sequence positions of the targeted jobs in the optimal sequence are identified, a number of alternate optimal solutions could be obtained by simple enumeration techniques. Alternate solutions give sequencing freedom for job operators to decide on the relative priority of different jobs. In the optimal sequence it is not necessary to list all the jobs in an order, but rather a few targeted jobs at the beginnings and ends of the sequences completely determine the completion time. More specifically, the first occurrences of the two jobs at the beginnings and ends of the optimal sequence for which the processing time on one machine attains its highest processing time on the other machine are very important because they completely determine the extent to which further computation of jobs to be assigned in the remaining sequence positions is no longer important as far as makespan is concerned. Therefore, Johnson's method was relaxed by terminating listing jobs at the first available positions when the job selected with the minimum processing time criteria on the first machine attains the highest processing time on the second machine for the first time and also by terminating listing jobs at the last available positions when the job with the minimum processing time criteria on the second machine attains the highest processing time on the first machine for the first time. The remaining jobs could be arranged in any convenient way in the remaining gaps without affecting minimum completion time.

Conflict of interest

The authors declare no conflict of interest.

References

1. Johnson, S.M. Optimal two and three-stages production schedules with set up time included, *Naval Resh. Logistic Quarterly,* 1954, **1(1)**, 61-68.
2. Ikram, M. On certain new terms in sequencing theory and on alternative optimal sequences. *Pure and Applied Mathematic Sciences*, 1975, 2, **1**, 45-50.
3. Prabha. "On alternate optimal solution procedure for solving nx2 flow-shop problem with transportation time and equivalent job-block", *Pure and Applied Mathematica Science,* 2014, **80**, 1-2.
4. Fora, L., Ikram, M. On Alternative Optimal Sequences for n-Jobs, Three Machines Scheduling Problem with Minimization of Total Elapsed Time, *International Journal of Science and Technology,* 2016, **6(4)**, ISSN 2224-3577.
5. Conway, R.W., Maxwell, W.L., Miller, L. W. *Theory of Scheduling,* Addison-Wesley, Reading, 1967.
6. Maggu, Alam and Ikram, M. On certain types of sequencing problems with n-jobs and m-machines, Pure and a Applied Mathematika Science, 1975, **1(2)**, 55-59.
7. Ancâu, M. On Solving flow-shop Scheduling Problems, *The Publishing House Proceedings of the Romanian Academy*, 2012, **13**, 71-79.
8. Palmer, D.S. "Sequencing jobs through a multi-stage process in the minimum total time-a quick method of obtaining a near optimum", *Operational Research Quarterly,* 1965, **16(1)**, 101-107.
9. Gupta, J.N.D. A functional Heuristic Algorithm for flow shop scheduling Problem, *Journal of Operational Research Society*, 1971, **22(1)**, 39-47.
10. Campbell, H.G., Dudek, R. A., Smith, M. L. A heuristic algorithm for the n job, m machine sequencing problem, *Management Science,* 1970, **16(10)**, 630-637.
11. Nawaz, M, Enscore Jr., E., Ham I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem, OMEGA, *The International Journal of Management Science,* 1983, **11(1)**, 91-95.
12. Koulamas, C. A new constructive heuristic for the flow-shop scheduling problem, *European Journal of Operational Research Society,* 1988, **105**, 66-71.
13. Dannenbring, D.G. An evaluation of flow shop sequencing heuristics, *Management Science,* 1977, **23(11)**, 1174-1182.
14. Neppalli, V.R.; Chen, C.L.; Gupta, J.N.D. Genetic algorithms for the two-stage bicriteria flow shop problem. *Eur. J. Oper. Res.*, 1996, **95**, 356-373.
15. Daniels, R.L., Chambers, R.J. Multi-objective flow shop scheduling, *Naval Research Logistics (NRL),* 1990, 37, **6**, 981-995.
16. Sayin, S., Karabati, S. Theory and methodology a bicriteria approach to the two-machines flow shop scheduling problem, *European Journal of Operational Research,* 1999, **113(2)**, doi: 10.1016/S0377-2217(98)00009-5.
17. Yeh, W.C. A new branch-and-bound approach for the $n/2/\text{flowshop}/\alpha F + \beta C_{max}$ flow shop scheduling problem, *Computers and Operations Research*, 1999, **26**(**13**), 1293-1310.
18. Yandra, H. Tamura. new multiobjective genetic algorithm with heterogeneous population for solving flow shop scheduling problems, *International Journal of Computer Integrated Manufacturing,* 2007, **20(5)**, 465-477.
19. Lebbar, G.A., El-Abbassi, Jabri, I.A., El Barkany, A., Darcherif, M. Multi-criteria blocking flow shop scheduling problems: Formulation and performance analysis, *Advances in Production Engineering and Management,* 2018, **13**, 136-146.

20. Radeleczki, S., T òth, T., G öndri-Nagy, J. A multiple (extended) application of the Johnson's algorithm for the two-machine manufacturing cell scheduling based on group technology, *Production Systems and Information Engineering, Miskolc*, 2003, **1**, 55-69.
21. Baker, K.R., Trietsch, D. *Principles of sequencing and scheduling,* Hoboken: Wiley, 2009.
22. Han, J., Kamber, M., Pei, J. *Data Mining Concepts and Techniques* (3ed.), Elsevier, 2012.
23. Maggu, P.L. and Das, G. "On 2xn sequencing problem with transportation times of Jobs", *Pure and' Applied Mathematika Sciences*, 1980, **12(1-2)**, 1-6.
24. Burges, C.J.C. Dimension Reduction: A Guided Tour, *Foundations and Trends in Machine Learning,* 2009, **2(4)**, 275-365.
25. Indhumathi, R., Sathiyabama, S. Reducing and Clustering high Dimensional Data through Principal Component Analysis, *International Journal of Computer Applications*, 2010, **11(8)**, 1-4.
26. Yan, J., Zhang, B., Liu, N., Yan, S., Cheng, Q., Fan, W., Yang, Q., Xi, W., Chen, Z. Effective and efficient dimensionality reduction for large-scale and streaming data preprocessing. *IEEE transactions on knowledge and data engineering,* 2006, **18(3)**, 320-332.

Received 5 February 2019
Accepted 30 October 2019