IHJPAS. 37 (1) 2024
Ibn Al-Haitham Journal for Pure and Applied Sciences
Journal homepage: jih.uobaghdad.edu.iq
PISSN: 1609-4042, EISSN: 2521-3407

Solving the Multi-criteria, Total Completion Time, Total Earliness Time, and Maximum Tardiness Problem

[1]Nagham Muosa Neamah **[2,*]Bayda Atiya Kalaf** **[3]Hamiden Abd El-Wahed Khalifa**

[1,2]Department of Mathematics, College of Education for Pure Sciences, Ibn Al – Haitham, University of Baghdad Iraq
[3]Department of Mathematics, College of Science and Arts, Qassim University, Al-Badaya 51951,Saudi Arabia.
[3]Department of Operations and Management Research, Faculty of Graduate Studies for Statistical Research, Cairo University, Giza 12613, Egypt

*Corresponding Author. baydaa.a.k@ihcoedu.uobaghdad.edu.iq

Received 13 December 2022, Received 6 January 2023, Accepted 13 March 2023, Published 20 January 2024

doi.org/10.30526/37.1.3094

Abstract

Machine scheduling problems (MSP) are considered as one of the most important classes of combinatorial optimization problems. In this paper, the problem of job scheduling on a single machine is studied to minimize the multi objective and multi objective function. This objective function is: total completion time, total lead time and maximum tardiness time, respectively, which are formulated as $\left(\sum \boldsymbol{C_j}, \sum \boldsymbol{E_j}, \boldsymbol{T_{max}}\right)$ are formulated. In this study, a mathematical model is created to solve the research problem. This problem can be divided into several sub-problems and simple algorithms have been found to find the solutions to these sub-problems and compare them with efficient solutions. For this problem, some rules that provide efficient solutions have been proved and some special cases have been introduced and proved since the problem is an NP-hard problem to find some efficient solutions that are efficient for the discussed problem 1// $F\left(\sum \boldsymbol{C_j}, \sum \boldsymbol{E_j}, \boldsymbol{T_{max}}\right)$, and good or optimal solutions for the multi-objective functions 1// $\sum C_j + \sum E_j + T_{max}$,, and emphasize the importance of the dominance rule (DR), which can be applied to this problem to improve efficient solutions.

Keywords: Maximum Tardiness, Multi-Criteria, Multi-Objective, Total Completion Times, Total Earliness Time.

1. Introduction

Scheduling involves distributing a set number of resources over a period of time to various tasks[1]. One or more objectives may be optimized as a result of this decision-making process. as well as, the Scheduling problem is defined as the arrangement of entities (people, tasks, vehicles, lecture, etc.) into a pattern in space-time in such a way that constraints are satisfied and certain goals are achieved [2-4].

Up until the late 1980s, mainstream research has concentrated on a certain single object problem. When more than one objective (criteria) is needed, scheduling problems become more difficult to model and solve. It is frequently implausible that different objectives will be best served by the same set of decision variables [5-8].

As a result, there is a trade-off between the multiple objectives. This type of problem is known as a multi-objective scheduling problem. Multi-objective scheduling problems are the term used to describe this kind of problem [9]. A set of Pareto optimal solutions (Efficient solutions), rather than a single optimal solution, are established using multi-criteria optimization based on competing objective functions. This set includes one (many) solution(s) that no other solution(s) is better with respect to objective functions[[10-14].

The most important literature survey for the last eight years. [15] discussed the multi-criteria in order to establish a collection of efficient solutions for the general problem, and scheduling problems that are researched on a single machine are considered. ***1//*** $\left(\sum C_j, \sum T_j, T_{Max}\right)$, ***1//*** $F\left(\sum C_j, \sum E_j, E_{Max}\right)$, ***1//*** $\sum C_j + \sum T_j + T_{Max}$, ***1//*** $\sum C_j + \sum E_j + T_{Max}$. [16] examined the multi-objective problem, which is the sum of completion time, tardiness, earliness, and late work. 1// $\sum_{=1}^{n}\left(E_j + T_j + C_j + U_j + V_j\right)$,1//$\sum_{=1}^{n}\left(\alpha_j E_j + \beta_j T_j + \theta_j C_j + \gamma_j U_j + \omega_j V_j\right)$,1/$S_f$/ $\sum_{=1}^{n}\left(\alpha_{jf} E_{jf} + \beta_{jf} T_{jf} + \theta_{jf} C_{jf} + \gamma_{jf} U_{jf} + \omega_{jf} V_{jf}\right)$. They suggested an Upper Bound (limits) UB and a Lower Boundary (limits) LB be used in the application of the Branch and Bound method. [17] studied the multi-criteria $\left(\sum C_j, T_{max}, R_L\right)$, multi-objective function $\left(\sum C_j + T_{max} + R_L\right)$ and founded the optimal solution by using the BAB method with and without DR then using some heuristic methods. [18] introduced a heuristic algorithm to reduce the $\left(\sum C_j + E_{max} + T_{max}\right)$ in a single machine scheduling.

In this paper, survey the tricriteria scheduling problem and begin with some basic scheduling concepts of multi-criteria problems, and basic rules are given in section 1. Section 2 provides information on problem formulation, analysis, and various algorithms. The Dominance Rule is described in section 3. In section 4 by proving several rules, we show there exists is an effective solution to our problem. The conclusions is given in section 5 and upcoming works.

2. Significant Notations.

In this paper, the following notations are used:

$\boldsymbol{N}$**:** jobs set $\text{s.t.}\, N = \{1,2,\dots,n\}$.

$\boldsymbol{n}$**:** number of available jobs.

$\boldsymbol{p_j}$**:** Ttime of the job $j's$ processing, which means it must be processed for a period of length $\boldsymbol{p_j}$.

$\boldsymbol{d_j}$**:** The due date for job j (or the jobs' due date), the optimal date for finishing the jobs*;* job termination after the deadline is allowed but will result in a penalty.

s_j: The Job's slack time for j s.t. $s_j = d_j - p_j$.
C_j: The job $j's$ completion time where $C_j = \sum_{k=1}^{j} p_k$.
L_j: The lateness time of jobs, s.t. $L_j = -(d_j - C_j) = C_j - d_j$.

E_j: The earliness of job j s.t. $E_j = max\{-L_j, 0\} = max\{d_j - C_j, 0\}$.
T_j: The tardiness of job j s.t. $T_j = max\{L_j, 0\} = max\{C_j - d_j, 0\}$.
$\sum C_j$: Total completion time.
$\sum E_j$: Total earliness time.
T_{max} : Maximum tardiness s.t. $T_{max} = max_{j\in N}\{T_j\}$.
F: The Ᵽ-problem's objective function.
F_1: The (SⱣ)-problem's objective function.
Shortest Processing Tim (*SPT*)*:* Jobs are Sequencing in non-decreasing order of the processing times p_j (i.e. $p_1 \le p_2 \le \cdots \le p_n$), this rule is well known to minimize $\sum C_j$ for problem 1// $\sum C_j$ [8].
Earliest Due Date (*EDD*)*:* Jobs are sequenced in non-decreasing order of their due dates d_j(i.e. $d_1 \le d_2 \le \cdots \le d_n$), this rule used to minimize T_{max} for problem 1// T_{max} [19].
Minimum Slack Time (*MST*)*:* Jobs are sequenced in non-decreasing order of their slack time $s_j = d_j - p_j$ (i.e. $s_1 \le s_2 \le \cdots \le s_n$). To minimize E_{max} using this rule [20].
Efficient Solution (*EFSO*): A feasible schedule α^* is known as Pareto optimal or (non-dominated) If there is absolutely no feasible schedule α, then the set of feasible schedules with regard to the criteria h_1 , h_2 and h_3 such that $h_1(\alpha) \le h_1(\alpha^*)$, $h_2(\alpha) \le h_2(\alpha^*)$ and $h_3(\alpha) \le h_3(\alpha^*)$, are satisfied with at least one of the inequalities [21].

3. Mathematical Formulation

In this section, the three-criteria scheduling problem (***1// $F(\sum C_j, \sum E_j, T_{max})$***) to be studied will be described. Let the number of jobs available at time 0 be represented by $N = \{1,2, \dots, n\}$, (i.e, $r_j = 0$ for all j) and need processing on just one machine. For each job, j has a processing time p_j and a due date d_j, given a list of jobs in the sequence $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$, the earliest completion time possible $C_j = \sum_{k=1}^{n} p_{\alpha_k}$, the tardiness of job j, $T_j = max\left\{C_j - d_{\alpha_j}, 0\right\}$, the earliness of job j, $E_j = max\left\{d_{\alpha_j} - C_j, 0\right\}$. The aim of this problem is finding a schedule $\alpha \in \mathcal{S}$ to find a schedule, (where $\mathcal{S}$ is the set of all possible feasible schedules; where a feasible schedule means it satisfies all the constraints of the problem Ᵽ) that minimizes the multi-criteria $(\sum C_j, \sum E_j, T_{max})$, which is denoted by $(SCSET)$, can be formulated mathematically as follows:

$$\left.\begin{array}{ll}
\multicolumn{2}{l}{\quad Min\{\sum C_j, \sum E_j, T_{max}\}} \\
\multicolumn{2}{l}{\quad \text{s.t.}} \\
C_1 = p_{\alpha_1} & \\
C_j \ge p_{\alpha_j} & j = 1,2, \dots, n \\
C_j = C_{\alpha_{(j-1)}} + p_{\alpha_j} & j = 2, \dots, n \\
T_j \ge C_j - d_{\alpha_j} & j = 1,2, \dots, n \\
E_j \ge d_{\alpha_j} - C_j & j = 1,2, \dots, n \\
T_j \ge 0, E_j \ge 0 & j = 1,2, \dots, n
\end{array}\right\} \qquad \dots(SCSET).$$

Where α_j indicate where job j falls in the ordering α and $\mathcal{S}$ represents the collection of all schedules. Finding the set of all efficient solutions to the problem $(SCSET)$ is challenging since it's an NP-hard problem (because the problem *1//* $\sum_{j=1}^{n} E_j$ is NP-hard [22]).

Proposition (1): There is an efficient schedule for the problem $(SCSET)$ that satisfies the SPT rule.

Proof: (a) first, assume that $p_i \neq p_j$ for all i, j. The unique sequence SPT, (SPT^*) provides the bare minimum of $\sum C_j$. As a result, no sequence exists $\delta \neq SPT^*$ s.t.

$$\sum C_j(\delta) \leq \sum C_j\,(SPT^*), \sum E_j\,(\delta) \leq \sum E_j\,(SPT^*), \text{and } T_{max}(\delta) \leq T_{max}(SPT^*) \quad (1)$$

The presence of one or more strict inequalities.

(b) If there is more than one sequence SPT (the processing times of jobs are equal), assume SPT^* be a sequence that satisfies the rule of SPT and jobs with equal processing times in the EDD and MST sequence. If a set of jobs that are to be early or partially early is specified, then this EDD and MST order minimized $\sum E_j$.

Note that if the event is several jobs at the same processing times, the due date is considered identical, or slack times, then SPT^* is not unique. Show that each SPT^*- sequence is an efficient, sequence that does not satisfy the SPT rule which cannot dominate an SPT^* sequence by (1.1). If δ is an SPT-sequences but not an SPT* sequence, it cannot dominate SPT^* since

$$\sum C_j(\delta) = \sum C_j(SPT^*), \sum E_j\,(SPT^*) \leq \sum E_j\,(\delta) \text{ and } T_{max}(SPT^*) \leq T_{max}(\delta)$$

by virtue of the EDD and MST rule. Hence all the SPT^* sequences are efficient.

As mentioned in proposition (1), shown that the SPT rule is efficient for the problem $(SCSET)$ but the EDD rule does not, as shown in the example below.

Example (1): Suppose the problem $(SCSET)$ has the following data in **Table 1**:

Table 1. The data of p_j, d_j, and s_j for problem $(SCSET)$

	Job1	*Job2*	*Job3*	*Job4*	*Job5*
p_j	2	5	7	5	8
d_j	6	9	8	11	14
s_j	4	4	1	6	6

A feasible schedule is provided by the SPT rule$(1,2,4,3,5)$ and $(1,4,2,3,5)$, hence $(\sum C_j, \sum E_j, T_{max}) = (67,6,13)$ from SPT^* order $(1,2,4,3,5)$ and $(\sum C_j, \sum E_j, T_{max}) = (67,7,13)$ from SPT order $(1,4,2,3,5)$, it is clear that in the SPT^*sequence the tasks $(2,4)$ are arranged with equal processing time in the rule of the MST or EDD. But EDD rule $(1,3,2,4,5)$ with $(\sum C_j, \sum E_j, T_{max}) = (71,4,13)$ and MST rule $(3,1,2,4,5)$ with $(\sum C_j, \sum E_j, T_{max}) = (76,1,13)$ hence $SPT^*$the sequence gives an efficient solution for the problem $(SCSET)$.

For the problem$(SCSET)$, we can deduce seven sub problems (SP_i) for $i = 1$ to 7:

1) *1//*$Lex(\sum C_j, \sum E_j, T_{max})$ problem $S₱_1$.
2) *1//*$Lex(\sum C_j, T_{max}, \sum E_j)$ problem $S₱_2$.
3) *1//*$Lex(T_{max}, \sum C_j, \sum E_j)$ problem $S₱_3$.
4) *1//*$Lex(T_{max}, \sum E_j, \sum C_j)$ problem $S₱_4$.
5) *1//*$Lex(\sum E_j, \sum C_j, T_{max})$ problem $S₱_5$.
6) *1//*$Lex(\sum E_j, T_{max}, \sum C_j)$ problem $S₱_6$.
7) *1//* $(\sum C_j + \sum E_j + T_{max})$ problem $S₱_7$.

(1) ***1//*** $\boldsymbol{Lex(\sum C_j, \sum E_j, T_{max})(S₱_1)}$***:*** The definition of this problem is as follows:

$$\left.\begin{array}{l} Min\ \{T_{max}\} \\ \text{s.t.} \\ \sum_{j=1}^{n} C_j = C^* \text{ where } C^* = \sum_{j=1}^{n} C_j(SPT) \\ \sum_{j=1}^{n} E_j \le E, E \in \left[\sum_{j=1}^{n} E_j\ (MST), \sum_{j=1}^{n} E_j\ (SPT)\right] \end{array}\right\} \qquad (S₱_1).$$

Since the most important function in this problem (S₱$_1$), $\sum_{j=1}^{n} C_j$, should be optimal, the following easy algorithm produces the best possible outcome.

Algorithm $(\boldsymbol{SCSET1})$ **for** ***1//***$\boldsymbol{Lex(\sum C_j, \sum E_j, T_{max})}$ **problem** $\boldsymbol{(S₱_1)}$**.**

ST1: is the Sequencing of jobs according to the SPT rule and the computation$(\sum \boldsymbol{C_j}, \sum \boldsymbol{E_j}, \boldsymbol{T_{max}})$. ***ST2:*** If there are jobs with equal processing times, then order these jobs: (a) using the MST rule and the calculation $(\sum \boldsymbol{C_j}, \sum \boldsymbol{E_j}, \boldsymbol{T_{max}})$. (b) using the EDD rule and the calculation$(\sum \boldsymbol{C_j}, \sum \boldsymbol{E_j}, \boldsymbol{T_{max}})$**.** ***ST3:*** If more than one SPT schedule appeared, then choose the schedule with minimum $\sum \boldsymbol{E_j}$ and $\boldsymbol{T_{max}}$.

Note: Same as an example (1) for $S₱_1$.

(2) ***1//***$\boldsymbol{Lex(\sum C_j, T_{max}, \sum E_j)(S₱_2)}$***:*** This problem is defined as follows:

$$\left.\begin{array}{l} Min\ \{\sum E_j\} \\ \text{s.t.} \\ \sum_{j=1}^{n} C_j = C^* \text{ where } C^* = \sum_{j=1}^{n} C_j(SPT) \\ T_{max} \le T, T \in [T_{max}(EDD), T_{max}(SPT)] \end{array}\right\} \qquad (S₱_2).$$

The problem (S₱$_2$)with $\sum_{j=1}^{n} C_j$ is the most important function, it must be optimal, so the easy **algorithm** $\boldsymbol{SCSET1}$ that gives us the best result for (S₱$_2$).

(3) ***1//***$\boldsymbol{Lex(T_{max}, \sum C_j, \sum E_j)(S₱_3)}$***:*** This problem is defined as follows:

$$\left.\begin{array}{l} Min \sum_{j=1}^{n} E_j \\ \text{s.t.} \\ T_{max} = T^* \text{ where } T^* = T_{max}(EDD) \\ \sum_{j=1}^{n} C_j \leq C, C \in [\sum_{j=1}^{n} C_j\,(SPT), \sum_{j=1}^{n} C_j(EDD)] \end{array}\right\} \quad (S₱_3).$$

Given that T_{max} is a more important function in this problem (S₱$_3$) and should be optimal, the following algorithm offers the best solution.

Algorithm ($SCSET2$) for $1//Lex(T_{max}, \sum C_j, \sum E_j)$ problem ($S₱_3$).

ST1: Arrange the jobs according to the rule of EDD and calculate $\boldsymbol{T_{max}(EDD) = T^*}$.
ST2: Calculated $D_i = d_i + T^*$, for all in N $where$ $N = \{1, \ldots, n\}$.
ST3: Suppose $t = \sum_{i \in N} p_i$ and K equals n.
ST4: Finding a job j using the Smith backward algorithm and satisfying $D_j \geq t$, $p_j \geq p_i$ (Choose the job j with the largest due date if there is a tie). Assign position $K$to job j.
ST5: Assign the variables $t = t - p_j$, $N = N - \{j\}$ and $K = K - 1$; if $K = 1$ proceed to step 6; if not, proceed to step 4.
ST6: Find $T_{max}, \sum C_j$ and $\sum E_j$ for the sequence that results.

Example (2): Consider the data for the problem $S₱_3$ in **Table 2.**

Table 2. The data of p_j, d_j , and s_j for problem $S₱_3$

	Job1	*Job2*	*Job3*	*Job4*
p_j	*9*	*3*	*7*	*2*
d_j	*12*	*5*	*9*	*10*
s_j	*3*	*2*	*2*	*8*

Hence the (EDD) schedule (2,3,4,1) gives $(T_{max}, \sum C_j, \sum E_j) = (46,2,9)$. $T_{max}(EDD) = T^*$, $t = 21$, $D_i = d_i + T^* = (21,14,18,19)$. The schedule (4,2,3,1) is given by Smith's backward algorithm with $(T_{max}, \sum C_j, \sum E_j) = (40,8,9)$.

(4) $\boldsymbol{1//Lex(T_{max}, \sum E_j, \sum C_j)(S₱_4)}$***:*** This problem is defined as follows:

$$\left.\begin{array}{l} Min \sum_{j=1}^{n} C_j \\ \text{s.t.} \\ T_{max} = T^* \text{ where } T^* = T_{max}(EDD) \\ \sum_{j=1}^{n} E_j = E, E \in [\sum_{j=1}^{n} E_j\,(MST), \sum_{j=1}^{n} E_j(EDD)] \end{array}\right\} \quad \ldots (S₱_4).$$

Given that T_{max} is a more important function in this problem S₱$_4$and should be perfect, the **algorithm $SCSET2$** offers the best solution.

(5) $\boldsymbol{1//Lex\left(\sum E_j, \sum C_j, T_{max}\right)}$ $(\boldsymbol{S₱}_5)$***:*** This problem is defined as follows:

$$\left.\begin{array}{l} Min\ \ \{T_{max}\ \} \\ \text{s.t.} \\ \sum_{j=1}^{n} E_j = E^* \ \text{ where } E^* \ = \sum_{j=1}^{n} E_j\ (MST) \\ \sum_{j=1}^{n} C_j \le C\ \ , C \in \left[\sum_{j=1}^{n} C_j(SPT)\,, \sum_{j=1}^{n} C_j\ (MST)\right] \end{array}\right\} \quad \ldots (S₱_5)\ .$$

(6) $\boldsymbol{1//Lex\left(\sum E_j, T_{max}, \sum C_j\right)}(\boldsymbol{S₱}_6)$***:*** The case can be written as:

$$\left.\begin{array}{l} Min\ \left\{\sum_{j=1}^{n} C_j\right\} \\ \text{s.t.} \\ \sum_{j=1}^{n} E_j = E^* \ \text{ where } \ E^* = min\left\{\sum_{j=1}^{n} E_j\ (MST)\right\} \\ T_{max} \le T\ , T \in [T_{max}(MST), T_{max}(EDD)] \end{array}\right\} \quad \ldots (S₱_6).$$

Given that *1//* $\sum E_j$ is an NP-hard problem, the problems $(S₱_5)$ and $(S₱_6)$ are both NP-hard.

(7) $\boldsymbol{1//\sum C_j + \sum E_j + T_{max}}$ **Problem.**

The objective of the problem is to find the sequence of job processing that will minimize $\sum C_j + \sum E_j + T_{max}$. Following is a definition of this sub-problem:

Suppose that α is any machine schedule that is possible to formulate as follows for a given schedule $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$. Assume that α is any schedule that can be expressed as follows for a certain schedule $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$:

$$\left.\begin{array}{ll} F_1 = Min\left\{\sum C_j + \sum E_j + T_{max}\right\} & \\ \text{s.t.} & \\ C_1 = p_{\alpha_1} & \\ C_j \ge p_{\alpha_j} & j = 1,2, \ldots, n \\ C_j = C_{\alpha_{(j-1)}} + p_{\alpha_j} & j = 2, \ldots, n \\ T_j \ge C_j - d_{\alpha_j} & j = 1,2, \ldots, n \\ E_j \ge d_{\alpha_j} - C_j & j = 1,2, \ldots, n \\ T_j \ge 0, E_j \ge 0 & j = 1,2, \ldots, n \end{array}\right\} \quad \ldots(S₱_7).$$

Finding a processing order $\alpha = (\alpha_1, \ldots, \alpha_n)$ for the jobs on a single machine that minimizes the sum of the total completion times, the total earliness, and the maximum tardiness $\left(\sum C_j(\alpha) + \sum E_j(\alpha) + T_{max}(\alpha)\right), \alpha \in \mathcal{S}$ (where $\mathcal{S}$ is the set of all feasible solutions), is the aim of this problem.

Proposition(2): The optimal solution for *1//* $\sum C_j + \sum E_j + T_{max}$ problem is an *EFSO* for the *1//* $F\left(\sum C_j, \sum E_j, T_{max}\right)$.

Proof: let β be an optimal schedule for *1//* $\sum C_j + \sum E_j + T_{max}$ problem. Suppose that β gives no efficient solution for the problem 1// $F\left(\sum C_j, \sum E_j, T_{max}\right)$, so there is a schedule α which is efficient for *1//* $F\left(\sum C_j, \sum E_j, T_{max}\right)$ problem such that:

$\sum C_j\ (\alpha) \le \sum C_j(\beta)$ and $\sum E_j\ (\alpha) \le \sum E_j(\beta)$ and $T_{max}(\alpha) \le T_{max}(\beta)$,

and when there are strict inequities in at least one. As a result, it follows:

$\sum C_j(\alpha) + \sum E_j(\alpha) + T_{max}(\alpha) \leq \sum C_j(\beta) + \sum E_j(\beta) + T_{max}(\beta)$, so, for*1//* $\sum C_j + \sum E_j + T_{max}$, α is a schedule that gives the better solution from β. However, since $\beta$is the optimal schedule, the assumption is contradicted, so β should give an efficient solution to *1//* $\sum C_j + \sum E_j + T_{max}$.

4. Special cases of the problems $(\boldsymbol{SCSET})$and $(\boldsymbol{S}₱_7)$

In this part, give some special cases and examples for problems $(\boldsymbol{SCSET})$and $(\boldsymbol{S}₱_7)$ that lead to efficient and optimal solutions respectively.

4.1 Special Cases of the Problem$(\boldsymbol{SCSET})$

Case$(\mathbf{4.1.1})$: If $p_1 = d_1$and $p_j = d_j - d_{j-1}$, for all j in α (except 1) then SPT schedule α gives an efficient schedule for problem $(SCSET)$**.**

Proof: Since $p_1 = d_1$ and $p_2 = d_2 - d_1 = d_2 - p_1$, then $C_1 = d_1$ and $C_2 = p_1 + p_2 = p_1 + d_2 - p_1 = d_2$ then $C_2 = d_2$ and so on $C_j = d_j$ for $j = 1,2,..,n$. Since $C_j = d_j$ for all j in σ hence $L_j = 0$, $\forall j$, then $E_j = T_j = 0$, so $\sum E_j = T_{max} = 0$.

Then the problem 1 // $\left(\sum C_j, \sum E_j, T_{max}\right)$ reduced to*1 //* $\sum C_j$.

But the rule that solved this problem was SPT.

Then α provides an efficient solution to $(SCSET)$ problem .

Case$(\mathbf{4.1.2})$: If $p_j = p$ and $d_j = jp$ for all j in the schedule σ,then σ gives an *EFSO* to $(SCSET)$.

Proof: Since $d_j = jp = C_j\ \forall j \in \sigma$, (this means there is no job late and early $\text{s.t.}\, E_j = 0 = T_j$) then $\sum_{j=1}^{n} E_j = T_{max} = 0$. Then the problem 1// $F\left(\sum C_j, \sum E_j, T_{max}\right)$ reduced to *1//* $\sum_{j=1}^{n} C_j$.

Now since $p_j = p$ for every job j in α, then $\sum_{j=1}^{n} C_j = p\left(\frac{n^2+n}{2}\right)$. But $p\left(\frac{n^2+n}{2}\right)$ is constant, hence any schedule gives an *EFSO* to $(SCSET)$.

As a result, every schedule provides an efficient solution to the problem.

Case $(\mathbf{4.1.3})$: If $d_j = kp_j$ for all $k \geq 2$ and $j \in$ α = SPT schedule then α is an *EFSO* to the $(SCSET)$.

proof: Let $s_j = d_j - p_j$ be the slack time of the job j $(j = 1, \dots, n)$ since $d_j = kp_j$ then $s_j = kp_j - p_j = (k-1)p_j$. Since SPT schedule, the processing time for tasks is arranged in a non-descending sequence (this meam $p_i \leq p_j$ for all $i \leq j$). Then $(k-1)p_1 \leq (k-1)p_2 \leq \cdots \leq (k-1)p_n$, hence $s_1 \leq s_2 \leq \cdots \leq s_n$. which is MST order, since MST order gives *EFSO* for $\sum E_j$. Hence SPT is efficient for $(SCSET)$**.**

Case$(\mathbf{4.1.4})$: If SPT and MST are identical then they give an efficient schedule for $(SCSET)$**.**

Proof: Since SPT and MST are identical then $\sum_{j=1}^{n} C_j$ is minimum value, and $\sum E_j$ is minimum value. But $s_j = d_j - p_j$ and $d_1 - p_1 \leq \ldots \leq d_n - p_n$ then $d_1 - p_1 + p_1 \leq \ldots \leq d_n - p_n + p_n$ (since $p_1 \leq \cdots \leq p_n$), hence $d_1 \leq \ldots \leq d_n$(which is EDD order), since the EDD order gives *EFSO* for the T_j then T_{max} is minimum. Hence all schedule an *EFSO* for $(SCSET)$.

Case$(\mathbf{4.1.5})$: If the processing times of all jobs are identical, then MST ordered is *EFSO to* problem $(SCSET)$.

Proof: Since the processing times of all jobs are identical, then $\sum C_j = p\left(\frac{n(n+1)}{2}\right)$, it is the same for any sequence. Since MST schedule, ordered the slack time of jobs in a non-decreasing sequence (that mean $s_i \leq s_j$ for all $i \leq j$ in MST schedule). Since $s_j = d_j - p$, using MST rule and adding p for each term, is produced $d_i \leq d_j$ for all $i \leq j$ (which is EDD order), hence EDD and MST are identical. Since MST order gives efficient value for $\sum E_j$ and EDD order gives efficient value for the T_j then T_{max} is minimum. Hence MST is an *EFSO* for the third criterion $\sum C_j, \sum E_j, T_{max}$.

Case$(4.1.6)$: If $d_j = d,$ and SPT and MST are identical $\forall j, (j = 1,2,\dots,n)$ in a schedule α, then α is *EFSO* to ($SCSET$).

Proof: Since $d_j = d$, there are two cases:

- **a)** If $d_j = d = p_j$, hence $C_j \geq d_j$ for all j (this means all jobs are late $\mathrm{s.t.} E_j = 0 = \sum E_j$ for all j). Then *1//* $\left(\sum C_j, \sum E_j, T_{max}\right)$ reduce to *1//* $\left(\sum C_j, T_{max}\right)$, as result, the SPT rule provides an *EFSO* to the problem ($SCSET$).where $d_j = d$ for all the orders of j and SPT gives *EFSO* for 1// $\sum C_j$.
- **b)** $d_j = d > p_j$ for all j hence either $d \leq C_j$ or $C_j > d$**,** since SPT and MST are identical (this mean $\sum C_j, \sum E_j$ is minimum values). Then a schedule α is an *EFSO* to ($SCSET$).

Case $(4.1.7)$: if $p_j = p, d_j = d$ and $d \leq C_j$ for all j in a schedule α then any schedule α is *EFSO* to ($SCSET$).

Proof: Since $d \leq C_j$ for all j (this means all jobs are late $\mathrm{s.t.} E_j = 0 = \sum E_j$) and $T_j = max\{L_j, 0\} = max\{jp - d, 0\}$ then $T_{max} = max\{max\{jp - d, 0\}\} = np - d$. Hence 1 // $\left(\sum C_j, \sum E_j, T_{max}\right)$ reduced to *1//* $\left(\sum C_j, T_{max}\right) = \left(p\left(\frac{n^2+n}{2}\right), np - d\right)$. Then any schedule is an *EFSO* to ($SCSET$) because the three quantities are constant.

Case$(4.1.8)$: If $d_{\alpha_i} + p_{\alpha_j} \leq d_{\alpha_j}$ for all i,j in SPT schedule α , where $i \leq j$, SPT and EDD are identical, then α is the *EFSO* for ($SCSET$).

Proof: Since $d_{\alpha_i} + p_{\alpha_j} \leq d_{\alpha_j}$ $\forall i,j$ in α, then $d_{\alpha_i} \leq d_{\alpha_j} - p_{\alpha_j}$ for all i,j and $d_{\alpha_i} - p_{\alpha_i} \leq d_{\alpha_j} - p_{\alpha_j}$, for all i,j (Since $p_{\alpha_i} \geq 0$ and $p_{\alpha_i} \leq p_{\alpha_j}$).Then $s_{\alpha_i} \leq s_{\alpha_j}$, for all i,j (this mean all jobs are ordered in MST order for all i,j in α), then $\sum E_j$ is minimum. Hence SPT gives the optimal solution for both criteria $\sum E_j$ *and* $\sum C_j$, and T_{max}(SPT) $=$ $T_{max}(EDD)$ (since SPT and EDD are identical). Then SPT is an *EFSO* for the problem ($SCSET$) .

Case$(4.1.9)$: If $d_j + p_j \leq C_{j+1}$ for $j = 1,2,..,n-1$ then the SPT schedule is an *EFSO* for ($SCSET$) .

Proof: Let $\sigma = (\sigma_1, \sigma_2, .., \sigma_n)$ by SPT sequence, since $d_j + p_j \leq C_{j+1}$, $\forall j (j = 1,2,..,n-1)$.

Then $d_j \leq C_{j+1} - p_j = \sum_{i=1}^{j+1} p_i - p_j = C_j + p_{j+1} - p_j$ $\forall j (j = 1,2,..,n-1)$.

$$C_j + p_{j+1} - p_j = \begin{Bmatrix} C_j & \text{if } p_j = p_{j+1} & ,j = 1,2,..,n-1 \\ C_j + p & \text{if } p = p_{j+1} - p_j & ,j = 1,2,..,n-1 \end{Bmatrix} \quad (1)$$

Since σ is the SPT schedule, there are two cases:

a) $p_j = p_{j+1}$, for $j = 1,2,..,n-1$ and Equation (1). Hence $d_j \le C_j$ (this means all jobs are late s. t. $E_j = 0 = \sum E_j$). The problem $1 // \left(\sum C_j, \sum E_j, T_{max}\right)$ reduced to *1//* $\left(\sum C_j, T_{max}\right)$, so SPT schedule is an *EFSO* .

b) $p_j < p_{j+1}$, for $j = 1,2,..,n-1$ and by (4.1),then $d_j \le C_j + p, p > 0$ then $d_j - C_j \le p$ (this means all jobs are late $s.t. E_j = 0 = \sum E_j$), hence $1 // \left(\sum C_j, \sum E_j, T_{max}\right)$reduced to *1//* $(\sum C_j, T_{max})$, also SPT rule is an *EFSO* .

Case$(\mathbf{4.1.10})$: If $C_j \ge d_j$ and SPT, EDD rules are identical for all j in a schedule α, then α schedule gives an *EFSO* for $(SCSET)$.

Proof: Let σ be an SPT schedule with $C_j \ge d_{\alpha_j}$ for each j in σ (this means that all jobs in the SPT schedule are late), hence $E_j = 0 = \sum E_j$ for all j in α. then 1// $F\left(\sum C_j, \sum E_j, T_{max}\right)$ reduced to 1//$\left(\sum C_j, T_{max}\right)$.Hence SPT schedule is efficient for $(SCSET)$.

Case(4.1.11): If the SPT schedule gives $C_j \le d_j\ \forall i \in N$ then this SPT schedule gives an *EFSO* for $(SCSET)$.

Proof: Since $C_j \le d_j$ for all j in the SPT schedule (this means all jobs are early s. t. $T_j = 0 = T_{max}$ for all j) and $E_j = max\{-L_j, 0\}$ then $\sum E_j = \sum max\{-L_j, 0\}$ for all j. Hence $1//$ $F\left(\sum C_j, \sum E_j, T_{max}\right)$ reduced to *1//* $(\sum C_j, \sum E_j)$. Hence SPT rule gives an *EFSO.*

4.2 Special Cases for Subproblem $(S\text{Ᵽ}_7)$

We introduce some special cases for the problem $(S\text{Ᵽ}_7)$that has optimal solutions in this section.

Case(4.2.1): If $p_1 = d_1$and $p_j = d_j - d_{j-1}$, for all j in α (except 1) then SPT schedule α gives an optimal solution for the problem 1// $\sum C_j + \sum E_j + T_{max}$**.**

Proof: Proof as in case (4.1.1) and $\left(\sum C_j + \sum E_j + T_{max}\right)$=$\sum C_j$.

Case(4.2.2): If $p_j = p$ and $d_j = jp$ for all j in the schedule σ, then σ gives an optimal solution for the problem $1// \left(\sum C_j + \sum E_j + T_{max}\right)$.

Proof: Verified by the case (4.1.2), hence $\left(\sum C_j + \sum E_j + T_{max}\right) = \sum_{j=1}^{n} C_j = p\left(\frac{n^2+n}{2}\right)$.

Case (4.2.3): If $d_j = kp_j$ for all $k \ge 2$ then the SPT schedule is an optimal solution for the problem $(S\text{Ᵽ}_7)$.

Proof: Proof as in case (4.1.3).

Case (4.2.4): If SPT and MST are identical then they give an optimal schedule for the problem $(S\text{Ᵽ}_7)$.

Proof: Proof as in case (4.1.4).

Case (4.2.5): If $p_j = p\ \forall j \in N$, then the EDD schedule is an optimal schedule for $(S\text{Ᵽ}_7)$.

Proof: Proof as in case (4.1.5).

Case(4.2.6): If $d_j = d,$ and SPT and MST are identical $\forall j, (j = 1,2,\dots,n)$ in schedule α then the α gives an optimal value for $(S\text{Ᵽ}_7)$**.**

Proof: Proof as in case (4.1.6), and

$$\sum_{j=1}^{n} C_j + \sum_{j=1}^{n} E_j + T_{max} = \begin{cases} \sum_{j=1}^{n} C_j + T_{max}, & \text{, if } d = p_j\left(i.e., C_j \geq d_j = d\right) \\ \sum_{j=1}^{n} C_j + \sum_{j=1}^{n} E_j + T_{max} = \sum_{j=1}^{n} d + T_{max}, & \text{if } d > p_j \text{ then either } C_j \geq d_j \text{ or } C_j < d_j \end{cases}.$$

Case (4.2.7): Any schedule gives an optimal solution for the problem S₱$_7$ if $p_j = p, d_j = d$ and $d \leq C_j\ \forall j (j = 1,2, \dots, n)$.

Proof: Proof as in case (4.1.7), and $\left(\sum C_j + \sum E_j + T_{max}\right) = p\left(\frac{n^2+n}{2}\right) + np - d$.

If $d_{\alpha_i} + p_{\alpha_j} \leq d_{\alpha_j}$ for all i, j in schedule $SPT\ \alpha$, where $i \leq j$ and SPT and EDD are identical, then α is the $EFSO$ for $(SCSET)$.

Case (4.2.8): If $d_{\alpha_i} + p_{\alpha_j} \leq d_{\alpha_j}$ for all i, j in schedule $SPT\ \alpha$, where $i \leq j$, SPT and EDD are identical, then α is the optimal solution for (S₱$_7$).

Proof: Proof as in case (4.1.8).

Case (4.2.9): If $d_j + p_j \leq C_{j+1}$ for $j = 1,2, .., n-1$ then the SPT schedule is an optimal solution for (S₱$_7$).

Proof: Verified by the case (4.1.9).

Case (4.2.10): If $C_j \geq d_j$, SPT and EDD rules are identical for all j in a schedule α, then schedule α gives an optmal solution for(S₱$_7$).

Proof: Verified by the case (4.1.10).

Case (4.2.11): If the SPT schedule gives $C_j \leq d_j\ \forall i \in N$ then this SPT schedule gives an optimal schedule for (S₱$_7$).

Proof: Proof as in case (4.1.11).

In Table 3, an examples give for describing the special cases for two problems ($SCSET$) and (S₱$_7$) by calculating the objective functions(F)and (F_1)respectively, using 6 jobs.

Table 3. Special Cases of Problem $(SCSET)$ and $(S\text{Ᵽ}_7)$ In the following examples**.**

case	p_j and d_j	conditions	F	F_1
(4.1.1) (4.2.1)	$p_j = 1,2,4,7,5,3$, $d_j =$ 1,3,10,22,22,15,6	$p_1 = d_1$ and $p_j = d_j - d_{j-1}$ for all j	(57,0,0)	57
(4.1.2) (4.2.2)	$p_j = 6, d_j = 6,12,18,24,30,36$	$p_j = p$ and $d_j = jp$, $\forall j$	(126,0,0)	126
(4.1.3) (42.3)	$p_j = 3,4,2,5,6,8$, $d_j =$ 9,12,6,15,18,24	$d_j = kp_j$ for all $k \geq 2$	(78,12,4)	94
(4.1.4) (4.2.4)	$p_j = 6,10,12,14,14,18$, $s_j =$ .2,4,8,10,12,13	for all j .$p_i \leq p_j$ and $s_i \leq s_j$	(222,2,43)	267
(4.1.5) (4.2.5)	$p_j = 3$, $d_j = 3,4,6,7,8,9$	.$p_j = p$ for all j in a schedule EDD α	(63,0,9)	72
(4.1.6)	$p_j = 5,4,3,2,1,1$, $d_j = 6$	$d_j = d, \forall j$	(41,11,10)	62
(4.2.6)	$p_j = 7,6,5,3,2,1$, $d_j = 7$		(106,0,17)	123
(4.1.7)	$p = d = 5$	$p_j = p$, $d_j = d, d \leq C_j$ for all j	(105,0,25)	130
(4.2.7)	$p = 5, d = 7$, $p < d$		(105,2,23)	130
(4.1.8) (4.2.8)	$p_j = 1,1,3,3,2,2$, $d_j = 1,2,9,12,4,6$	, SPT and $d_i + p_j \leq d_j$ for all j EDD rules are identical.	(34,0,0)	34
(4.1.9) (4.2.9)	$p_j = 5,4,3,2,4,2$, $d_j = 6,7,4,3,5,2$	$d_j + p_j \leq C_{j+1}$ for all $j = 2,3,\ldots,n$	(59,0,14)	73
(4.1.10)	$p_j = 4,3,2,2,1,1$, $d_j = 14,9,5,6,2,3$	$C_j \geq d_j$, for all j	(35,0,4)	39
(4.2.10)	$p_j = 3,4,5,6,8,9$, $d_j = 3,5,6,7,9,10$		(101,0,25)	126
(4.1.11) (4.2.11)	$p_j = 8,5,2,6,4,3, d_j = 30,14,2,21,10,6$	$C_j \leq d_j$, for all j	(78,5,0)	83

5. Dominance Rules for Single Machine Scheduling Problem

Dominance Rules (DRs) are used efficiently in reducing the current sequences[23-25]. DR is used usually to indicate whether a certain node in a BAB method can be eliminated before calculating its lower bound. These rules are been useful when a node has a lower bound less than the optimum solution and can be eliminated [26-28]. When the nodes are dominated by others in the BAB procedure, DRs can be used also to cut these nodes. Such developments may heavily reduce the number of nodes in searching for an efficient solution. Where the DRs are also applicable to such problems[29,30].

The dominance rules as we mentioned before are used in an attempt to eliminate nodes in the BAB method which makes us reduce the time spent on solving this problem$1//\sum C_j + \sum E_j + T_{max}$.

Rule (1):

If $p_i \leq p_j$ and $d_i \leq d_j$ then there is an optimal schedule wherein the job i processing before job .

Proof: Consider a schedule $\sigma = \sigma_1 ij\sigma_2$ and a schedule $\dot{\sigma} = \sigma_1 ji\sigma_2$ see **Figure (1)**

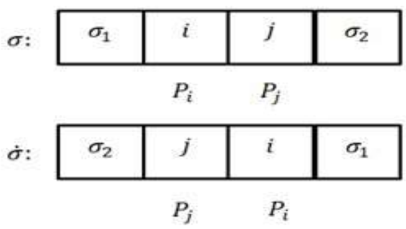

Figure1. The scheduling σ and$\ddot{\sigma}$

which is obtained by interchanging the jobs i and j in σ. For these schedules, we study two cases, and in every case, we will make a comparison between them.

First case: If $p_i \leq p_j$, $d_i \leq d_j$ produces that $s_i \leq s_j$

In this situation, there are:

The condition of the processing times ensures that:

$\sum C_{\mathcal{K}}(\sigma) \leq \sum C_{\mathcal{K}}(\dot{\sigma})$ (2)

And the condition of the slack times ensures that:

$E_{max}(\sigma) \leq E_{max}(\dot{\sigma})$ then $\sum E_{\mathcal{K}}(\sigma) \leq \sum E_{\mathcal{K}}(\dot{\sigma})$

And the condition on the due date ensures that:

$T_{max}(\sigma) \leq T_{max}(\dot{\sigma})$ (3)

Hence $\sum C_{\mathcal{K}}(\sigma) + \sum E_{\mathcal{K}}(\sigma) + T_{max}(\sigma) \leq \sum C_{\mathcal{K}}(\dot{\sigma}) + \sum E_{\mathcal{K}}(\dot{\sigma}) + T_{max}(\dot{\sigma})$

Second case: If $p_i \leq p_j$, $d_i \leq d_j$ yields that $s_i \geq s_j$.

In this situation, The condition on the processing times ensures that (5.1) is satisfied, and the cost which is obtained from Equation (2) is equal to $p_j - p_i$

i.e. $\sum C_{\mathcal{K}}(\dot{\sigma}) = \sum C_{\mathcal{K}}(\sigma) + p_j - p_i$ (4)

, then $d_i - C_i(\sigma) = d_i - C_j(\dot{\sigma}) + p_j - p_i$, since $C_i(\sigma) = C_j(\dot{\sigma}) - p_j + p_i$.

since $s_i = d_i - p_i \geq s_j = d_j - p_j$ then $d_i - p_i - C_j(\dot{\sigma}) \geq d_j - p_j - C_j(\dot{\sigma})$ hence $d_i - p_i + p_j - C_j(\dot{\sigma}) \geq d_j - C_j(\dot{\sigma})$

, from which we deduce that $E_{max}(\sigma) \geq E_{max}(\dot{\sigma})$ then $\sum E_{\mathcal{K}}(\sigma) \geq \sum E_{\mathcal{K}}(\dot{\sigma})$ the addition in cost which is obtained from this inequality is equal to $s_i - s_j$, i.e.,

. $\sum E_{\mathcal{K}}(\sigma) = \sum E_{\mathcal{K}}(\dot{\sigma}) + (s_i - s_j)$ (5)

Since $p_i \leq p_j$ then $p_j - p_i \geq 0 \quad \forall i, j$

Since $d_i \leq d_j$ then $d_j - d_i \geq 0 \quad \forall i, j$

From $s_i - s_j \leq p_j - p_i$, then $\sum E_{\mathcal{K}}(\dot{\sigma}) + (s_i - s_j) \leq \sum E_{\mathcal{K}}(\dot{\sigma}) + p_j - p_i$.

(by adding $\sum E_k(\dot{\sigma})$ for both sides) by Equation (5)

Then $\sum C_{\mathcal{K}}(\sigma) + \sum E_{\mathcal{K}}(\sigma) \leq \sum C_{\mathcal{K}}(\sigma) + p_j - p_i + \sum E_{\mathcal{K}}(\dot{\sigma})$.

From Equation (3)

$\sum C_{\mathcal{K}}(\sigma) + \sum E_{\mathcal{K}}(\sigma) \leq \sum C_{\mathcal{K}}(\dot{\sigma}) + \sum E_{\mathcal{K}}(\dot{\sigma})$(by adding T_{max} for both sides) .

And $\sum C_{\mathcal{K}}(\sigma) + \sum E_{\mathcal{K}}(\sigma) + T_{max}(\sigma) \leq \sum C_{\mathcal{K}}(\dot{\sigma}) + \sum E_{\mathcal{K}}(\dot{\sigma}) + T_{max}(\dot{\sigma})$.

Rule (2):

The schedule σij is dominated by the schedule σji if the following inequalities hold:

(1) $p_i \leq p_j$ *(2)* $d_j \leq d_i$ *(3)* $d_j \geq s_i$.

Proof: Consider a schedule $\sigma = \sigma_1 ij\sigma_2$ and a schedule $\dot{\sigma} = \sigma_1 ji\sigma_2$

which is obtained by interchanging the jobs i and j in σ. The condition of the processing times ensures that ($\sum C_{\mathcal{K}}(\sigma) \leq \sum C_{\mathcal{K}}(\dot{\sigma})$

is satisfied and then the addition in cost is obtained by

($\sum C_{\mathcal{K}}(\dot{\sigma}) = \sum C_{\mathcal{K}}(\sigma) + p_j - p_i$ (6)

Since $C_i(\sigma) = C_j(\dot{\sigma}) - p_j + p_i$ then $d_i - C_i(\sigma) = d_i - C_j(\dot{\sigma}) + p_j - p_i$.The condition (1), (2) implies, since $s_i \geq s_j$ then $d_i - p_i - C_j(\dot{\sigma}) \geq d_j - p_j - C_j(\dot{\sigma})$ hence $d_i - p_i + p_j - C_j(\dot{\sigma}) \geq d_j - C_j(\dot{\sigma})$ from which we deduce that $E_{max}(\sigma) \geq E_{max}(\dot{\sigma})$ then $\sum E_{\mathcal{K}}(\sigma) \geq \sum E_{\mathcal{K}}(\dot{\sigma})$.

Then the addition in cost from this inequality is obtained from

$(\sum E(\sigma) = \sum E_{\mathcal{K}}(\dot{\sigma}) + (s_i - s_j))$

, hence $d_i - d_j + p_j - p_i \geq p_j - p_i$, and $d_i \geq d_j$ and $p_j \geq p_i$ this mean $p_j - p_i \geq 0$

. Then $\sum E_k(\dot{\sigma}) + (s_i - s_j) \geq \sum E_{\mathcal{K}}(\dot{\sigma}) + p_j - p_i$, (by adding $\sum E_{\mathcal{K}}(\dot{\sigma})$ for both $s_i - s_j \geq p_j - p_i$ sides).

from ($\sum E_{\mathcal{K}}(\sigma) = \sum E_{\mathcal{K}}(\dot{\sigma}) + (s_i - s_j)$)

and then $\sum C_{\mathcal{K}}(\sigma) + \sum E_{\mathcal{K}}(\sigma) \geq \sum C_{\mathcal{K}}(\sigma) + p_j - p_i + \sum E_{\mathcal{K}}(\dot{\sigma})$, (by adding $\sum C_k(\sigma)$ for both sides).

From Equation (6) there is $\sum C_{\mathcal{K}}(\sigma) + \sum E_{\mathcal{K}}(\sigma) \leq \sum C_{\mathcal{K}}(\dot{\sigma}) + \sum E_{\mathcal{K}}(\dot{\sigma})$ (by adding T_{max} for both sides), finally $\sum C_{\mathcal{K}}(\sigma) + \sum E_{\mathcal{K}}(\sigma) + T_{max}(\sigma) \leq \sum C_k(\dot{\sigma}) + \sum E_{\mathcal{K}}(\dot{\sigma}) + T(\dot{\sigma})$.

Example (3): Let's use MSP with 5 jobs and processing time and due date as the following table:

Table 4. The data of p_j, d_j , and s_j for problem $(SCSET)$

	1	*2*	*3*	*4*	*5*
p_j	1	8	10	4	9
d_j	14	28	27	23	12
s_j	13	20	17	19	3

When using the Rule (1) we obtain the DRs mentioned in **Figure (3).**

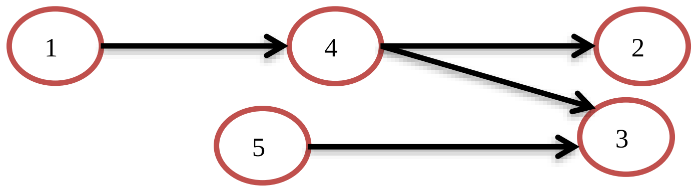

Figure (3): The DRs of the example (3).

From Rule (1) there are (6) DRs, as can see: 1→2, 1→ 3,1→4, 1→6, 2→ 6, 4→2, 4→ 3, 4→ 6,5→3. in **Table 5**, contain (7) likely sequences some /all are subject to the aforementioned DRs. The adjacency matrix A is as followings:

$$A(G) = \begin{bmatrix} 0 & 1 & 1 & 1 & a_{15} \\ 0 & 0 & a_{23} & 0 & a_{25} \\ 0 & a_{32} & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & a_{45} \\ a_{51} & a_{52} & 1 & a_{54} & 0 \end{bmatrix}. \text{ where } a_{ji} = \begin{cases} 1, \text{if } a_{ij} = 0 \\ 0, \text{if } a_{ij} = 1 \end{cases} .$$

Table 5. The efficient sequences for example (3) under DR

Seq.	***EF-SQ*** ***POS1***	***POS 2***	***POS 3***	***POS 4***	***POS 5***	$(SCSET)$ $(\sum C_j, \sum E_j, T_{max})$	$S₱_7$ $\sum C_j + \sum E_j + T_{max}$
1	1	4	2	5	3	(73,46,10)	129
2	1	4	5	2	3	(74,37,5)	116
3	1	4	5	3	2	(76,34,4)	114
4	1	5	4	2	3	(79,30,5)	114
5	1	5	4	3	2	(81,27,5)	113
6	5	1	4	2	3	(87,22,4)	114
7	5	1	4	3	2	(89,19,4)	112

Where ***EF-SQ*** =efficient sequence, ***POS*** = position

The sequences (1–7) provide the problem $(SCSET)$ an effective value and the sequence (7) an optimal value for the problem ($S₱_7$), as can be shown in **Table 5**.

6. Conclusions

In this study, a mathematical model was created to address the research problems $1//F(\sum C_j, \sum E_j, T_{max})$, $1// \sum C_j + \sum E_j + T_{max}$. Discovered a straightforward algorithm for the efficient schedule for sub-problems for $1//F(\sum C_j, \sum E_j, T_{max})$ and it was proven that some rules give efficient (optimal) solutions to these problems $(SCSET)$ and ($S₱_7$), finding and proving some special cases that find some efficient (optimal) solutions suitable for the problems $(SCSET)$ and ($S₱_7$). This paper has proved the efficacy of rules SPT and EDD and demonstrated the significance of the Dominance Rule (DR) that can be used in this problem to improve efficient solutions.

In the future, it would be interesting to conduct a study on the following machine scheduling problems (MSPs).

1) $1/r_j/F(\sum C_j, \sum E_j, T_{max})$.
2) $1/r_j/\sum C_j + \sum E_j + T_{max}$.
3) $1/S_f/F(\sum C_j, \sum E_j, T_{max})$.

Acknowledgment

The authors are greatly appreciated the referees for their valuable comments and suggestions for improving the paper

Conflict of Interest

The authors declare that they have no conflicts of interest.

Funding

There is no financial support in preparation for the publication.

References

1. WASSENHOVE, L.N.V. Single Machine Scheduling to Minimize Total Late Work. *Oper. Res.* **1992**, *40*, 586–595. https://doi:10.1287/opre.40.3.586.
2. Tanaev, V.; Gordon, W.; Shafransky, Y. M. *Scheduling theory*. *Single-stage systems*. Springer Science and Business Media. **2012.**
3. Hoogeveen, H. Multicriteria Scheduling. Eur. J. *Oper. Res.* **2005**, *167*, 592–623. https://doi:10.1016/j.ejor.2004.07.011.

4. Zaidan, A.A.; Atiya, B.; Abu Bakar, M.R.; Zaidan, B. B. A new hybrid algorithm of simulated annealing and simplex downhill for solving multiple-objective aggregate production planning on fuzzy environment. *Neural Comput & Applic*, **2019**, *31*, 1823–1834. https://doi.org/10.1007/s00521-017-3159-5
5. Chachan, H. A., ; Jaafar, H. A. Exact Solutions for Minimizing cost Function with Five Criteria and Release Dates on Single Machine. *Ibn AL-Haitham Journal For Pure and Applied Sciences*, **2020**. *33(3)*, 140–157. https://doi.org/10.30526/33.3.2479
6. Ali, Z.M.; Abdul Razaq, T.S. Minimizing The Total Completion Times, The Total Tardinessand The Maximum Tardiness. **2015**, *28*, 155–170.
7. Ahmed, M.G.; Ali, F.H. Exact Method with Dominance Rules for Solving Scheduling on a Single Machine Problem with Mult iobjective Function. *Al-Mustansiriyah J. Sci*. **2022**, *33*, 56–63. https://doi:10.23851/mjs.v33i2.1091.
8. Ali, F.H.; Ahmed, M.G. Local Search Methods for Solving Total Completion Times, Range of Lateness and Maximum Tardiness Problem. Proc. 6th Int. Eng. Conf. Sustainable Technol. Dev. IEC **2020**, 103–108. https://doi:10.1109/IEC49899.2020.9122821.
9. Hassan, D.A.; Mehdavi-Amiri, N.; Ramadan, A.M. A Heuristic Approach to Minimize Three Criteria Using Efficient Solutions. *Indones. J. Electr. Eng. Comput. Sci.* **2022**, *6*, 334–341. https://doi:10.11591/ijeecs.v26.i1.pp334-341.
10. Abdullah, H. F. Multicriteria Scheduling Problems to Minimize Total Tardiness Subject to Maximum Earliness or Tardiness. *Ibn AL-Haitham Journal For Pure and Applied Sciences*, **2017,** *23(1)*, 311–320. https://jih.uobaghdad.edu.iq/index.php/j/article/view/988
11. Smith, W.E. Various Optimizers for Single-Stage Production. *Nav. Res. Logist*. Q. **1956**, *3*, 59–66, https://doi:10.1002/nav.3800030106
12. Abdul-Razaq, T.S.; Akram, A.O. Local Search Algorithms for Multi-Criteria Single Machine Scheduling Problem. *Ibn AL-Haitham J. Pure Appl. Sci***.** **2018**, 436–451.https://doi:10.30526/2017.ihsciconf.1817.
13. Hoogeveen, J.A. Minimizing Maximum Promptness and Maximum Lateness on a Single Machine. *Math. Oper. Res*. **1996**, *21*, 100–114, https://doi:10.1287/moor.21.1.100.
14. Jawad, A.A.; Ali, F.H.; Hasanain, W.S. Using Heuristic and Branch and Bound Methods to Solve a Multi-Criteria Machine Scheduling Problem. *Iraqi J. Sci*. **2020**, *61*, 2055–2069. https://doi:10.24996/ijs.2020.61.8.21
15. Abdul-Zahra, I.; Abbas, I. T.; Kalaf, B. A.; Bakar, R. A.; June, L. W., ; Monsi, M. B. . The Role of Dynamic Programming in the Distribution of Investment Allocations between Production Lines with an Application. *International Journal of Pure and Applied Mathematics*, **2016**, *106(2)*, 365-380.
16. Atiya, B.; Bakheet, A. J. K.; Abbas, I. T.; Bakar, M. R. A.; Soon, L. L., ; Monsi, M. B. Application of simulated annealing to solve multi-objectives for aggregate production planning. AIP Conference Proceedings (2016, June). *1739(1)*, 020086). AIP Publishing LLC. https://doi.org/10.1063/1.4952566
17. Kalaf, B. A., Mohammed, G. J.; Salman, M. D. A New Hybrid Meta-Heuristics Algorithms to Solve APP Problems. In *Journal of Physics: Conference Series* (2021, May). *1897(1)*, 012011). IOP Publishing.
18. Allahverdi, A.;Ng, C. T.; Cheng, T. E.; Kovalyov, M. Y. A survey of scheduling problems with setup times or costs, *European Journal of Operational Research*, **2008**. *187(3)*, 985–1032. https://doi.org/10.1016/j.ejor.2006.06.060.
19. Allaoua, H.; Brahim, B. New Properties for Solving the Single-Machine Scheduling Problem with Early/Tardy Jobs, *Journal of Intelligent Systems***, 2017,** *26(3)*, 531–543. https://doi.org/10.1515/jisys-2016-0063.
20. Jawad, A.A.; Ali, F.H. ; Hasanain, W.S. .Using heuristic and branch and bound methods to solve a multi-criteria machine scheduling problem, *Iraqi Journal of Science***, 2020**, *61(8)*, 2055–2069. https://doi.org/10.24996/ijs.2020.61.8.21

21. Mahnam, M.; Moslehi, G.; Fatemi Ghomi, S.M.T. Single machine scheduling with unequal release times and idle insert for minimizing the sum of maximum earliness and tardiness, *Mathematical and Computer Modelling*, **2013,** *57(9–10)*, 2549–2563 https://doi.org/10.1016/j.mcm.2013.01.007.
22. Mosheiov, G.; Oron, D. ; Shabtay, D. Minimizing total late work on a single machine with generalized due-dates, *European Journal of Operational Research*, **2021,** *293(3),* 837–846. https://doi.org/10.1016/j.ejor.2020.12.061.
23. Neamah, N. M.; Kalaf, B. A. Solving the multi-criteria: Total completion time, total late work, and maximum earliness problem. *Periodicals of Engineering and Natural Sciences*, **2023.** *11*(*3*), 46-57.
24. Centinkaya, M.; Ozyurek, C., The Effect of Inquiry-Based Science Activities on Prospective Science Teachers' Scientific Process Skills, *International Online Journal of Education and Teaching*, . **2019,** *6(1)*, 56–70.
25. Chachan, H.A. Solving Machine Scheduling Problem Using Particle Swarm Optimization Method, *The Iraqi Magazine For Administrative Sciences*, **2012,** *8(33)*, 197–213.
26. Abid, H. ; Mohammed, A. Scheduling job families with setups on a single machine, *Journal of Kerbala University*, **2012**, *10(2)*, 99–113.
27. Adel D., Search method for solving multicriteria scheduling problem, **2022**, 13(March 2021), 1709–1720.
28. Ahmadov, Y. ; Helo, P., A cloud based job sequencing with sequence-dependent setup for sheet metal manufacturing', Annals of Operations Research, **2018**, *270(1–2)*,5–24. https://doi.org/10.1007/s10479-016-2304-3.
29. Ahmed, M. ; Ali, F. *Efficient Algorithms to Solve Tricriteria Machine Scheduling Problem*. *Journal of Al-Rafidain University College For Sciences*, **2020,** *(1)*, 485-493.
30. Ahmed, M.G. and Ali, F.H. Exact Method with Dominance Rules for Solving Scheduling on a Single Machine Problem with Multi objective Function', *Al-Mustansiriyah Journal of Science*, **2022,** *33(2)*, 56–63. Available at: https://doi.org/10.23851/mjs.v33i2.1091.