International Journal of Production Management and Engineering

https://doi.org/10.4995/ijpme.2024.19277
Received: 2023-02-16 Accepted: 2023-07-29

Hybrid genetic algorithm to minimize scheduling cost with unequal and job dependent earliness tardiness cost

Prasad Bari [a1], **Prasad Karande** [b], **Vaidehi Bag** [a2]

[a]Rodrigues Institute of Technology, 400703, Vashi, Navi-Mumbai, India.
[b]Department of Mechanical Engineering, Veermata Jijabai Technological Institute, 400019, Mumbai, India .
sainc17@centurylink.net

Abstract:

This article presents two combinatorial genetic algorithms (GA), unequal earliness tardiness-GA (UET-GA) and job-dependent earliness tardiness-GA (JDET-GA) for the single-machine scheduling problem to minimize earliness tardiness (ET) cost. The sequence of jobs produced in basic UET and JDET is added to the random population of GA. The best sequence from each epoch is also injected as a population member in the subsequent epoch of GA. The proposed improvement seeks to achieve convergence in less time to search for an optimal solution. Although the GA has been implemented very successfully on many different types of optimization problems, it has been learnt that the algorithm has a search ability difficulty that makes computations NP-hard for types of optimization problems, such as permutation-based optimization problems. The use of a plain random population initialization results in this flaw. The main objective of the article is to develop the combination of heuristics (UET, JDET) and the GA to obtain convergence by reinforcing the random population initialization and finding a promising solution for the reduction in total ET cost. The cost is further significantly lowered offering the due date as a decision variable with JDET-GA. Multiple tests were run on well-known single-machine benchmark examples to demonstrate the efficacy of the proposed methodology, and the results are displayed by comparing them with the fundamental UET and JDET approaches with a notable improvement in cost reduction.

Key words:

Earliness, Tardiness, Cost, Common Due Date, Genetic Algorithm.

1. Introduction

Over the last few decades, operational research practitioners have paid close attention to scheduling issues. For many years, however, great emphasis was placed on regular performance measurements, which are based on job tardiness, and total flow time. Oyetunji (2009) discussed performance measurements related to tardiness and total flow time so that their values could be calculated. Gupta and Kumar (2016) used the TOPSIS approach for flow shop scheduling problems to minimize the makespan criterion which is related to total flow time performance measure. Performance measures concerning tardiness are met by meeting due dates. Bari et al. (2022b) studied flow time-related measures as well as tardiness-related measures, and they used a genetic algorithm (GA) to determine the optimal solution for minimizing tardiness measures. Manufacturers must make commitments to meet promised due dates and complete jobs ahead of schedule. However, manufacturers do not want to produce too early. When a job is finished ahead of schedule, it is referred to as an early job. An early job is undesired since it requires warehouse storage, which results in inventory and insurance fees. When a job is finished later than the assured due date, it is referred to as a tardy job. A tardy job is undesirable since the manufacturer misses some of its income as well as some goodwill and reputation due to contract penalties. The rise of the just-in-time (JIT) mindset emphasised the significance of evaluating the cost configurations related to both early and tardy jobs, resulting in the rapid spread of non-regular performance measures (Rolim and Nagano, 2020). Many researchers (Shabtay and Steiner, 2007; Kanet and Sridharan, 2000; Woodruff and Spearman, 1992; Ow and Morton, 1989) mentioned that the goal of optimizing total earliness and tardiness in scheduling is stable with a JIT concept in which it is preferable to finish the jobs as near to their due dates as possible. For more than three decades, researchers have been studying the earliness tardiness (ET) scheduling problem with a common due date or distinct due date. The problem is a scheduling problem with a common due date if all jobs have the same deadline. If the jobs have arbitrary due dates, then the problem is a distinct due date scheduling problem. A common due date can be considered as a given constraint or choice variable in the ET schedule. Due date is issued in advance in the context of frequent circumstances where factory floor actions must satisfy client needs, that are impacted by outside sources. Treating them as decision variables, on the other hand, may convert production systems that set goals to direct the completion of inside tasks. Therefore, the due date is either given or regarded as a decision variable. If the due date is known to manufacturers then the cost function is objective, on the other hand, if the due date is uncertain then the due date is an optimization criterion for manufacturers.

Arık et al., (2022) studied that usually, there is no downtime between any two consecutive jobs to generate an optimum sequence with minimum cost. When jobs have a common due date, the issue of no downtime among jobs is usually inevitably fulfilled, excluding at the start of the schedule. When job due dates are distinct, incorporating downtime among jobs can be advantageous (Baker and Scudder, 1990). But operating a machine inactively is either prohibitively expensive or technologically limited, and hence jobs must be completed with no idle time in between (Chen et al., 2012; Carlier et al., 2010; M'Hallah, 2007; Irani and Pruhs, 2005; Wagner et al., 2002; Liaw, 1999). There are numerous real-world scenarios in which the supposition of no idle time between jobs is reasonable. With known and common due dates, a scheduling problem is classified as a restrictive and unrestrictive scheduling problem to optimize the cost function. The optimum sequence of jobs on a single machine ET scheduling problem is V-shaped (Kanet, 1981). The V-shaped feature demonstrates that jobs that are finished early are scheduled in the highest to the lowest order of processing time, whereas jobs that are finished late are scheduled in rising order of processing times.

Akande and Ajisegiri, (2022) discussed three distinct tolerance windows, the positive tolerance window, in which there is a profit with early jobs, as in some industrial systems where demand surpasses supply. The negative tolerance window, where early jobs are penalized, particularly as a function of inventory cost. The window of neutral tolerance where the job's first arrival invites neither advantage nor penalty. Based on the tolerance window, the ET problem can be classified into three classes, the first benefits from early jobs and penalties for jobs that are late. The second class is the one where tardy jobs and early jobs both carry penalties. JIT is incorporated into this model that specifies, the job's completion time should coincide with its due date. The third class has no advantage or penalty linked with previous jobs, but there is a penalty linked with delayed jobs.

The majority of scheduling problems are NP-hard. Integer or mixed integer programming (MIP) are common methodologies for optimizing or heuristically addressing these issues. Total ET scheduling, in particular, are NP-hard in the strict sense. When the downtime between jobs is permitted several investigators have suggested MIP formulations (Alidaee et al., 2021), but this may not be preferred many a time.

The GA is mostly used to minimize the penalties for being early and tardy in the single-machine ET problem. Schaller and Valente (2013) suggested a GA for minimizing total earliness and tardiness and compared it to neighborhood search techniques and metaheuristics. Although the GA is employed solely in literature to address this topic, it is challenging to conclude the optimal solution because of the algorithm's randomness. The performance of the method can be enhanced by integrating other algorithms, especially to enhance the local search principle (Yuce et al., 2017). For single-machine scheduling of a group of jobs with a common due date, they developed a meta-heuristic to reduce the job's overall earliness and tardiness. They created a hybrid genetic bees' algorithm by fusing the exploration capabilities of the GAs with the exploitation capabilities of the bees' algorithm. Plateau and Rios-Solis (2010) presented a quadratic programming approach for the problem of a common due date constraint to optimize the cost function. Beyranvand et al. (2012) further improved this approach for the same problem. For the problem, Arık (2020) proposed simulated annealing (SA), an artificial bee colony (ABC), and a GA to optimize the solution.

Dynamic programming-based GA was proposed by Allaoua and Osmane (2010), where the population size and chromosome of the solution grow as the number of iterations rises. Their objective was to reduce the total ET penalties in the schedule of jobs for a single machine. Raghavan et al., (2018) developed a modified GA to address the scheduling problem with stochastic rework and reprocessing time in the electronics industry. The objective was to reduce the total weighted tardiness. The goal of Khoshjahan et al. (2013) was to reduce the net present value of the ET penalty charges by taking into account the resource-constrained project scheduling challenge. They applied GA and SA, fine-tuning their parameters using response surface methodology. Their computational results showed that SA does superior to the GA in terms of accuracy index, but seeing computation time taking place on a computer the GA is better. Stanković et al. (2020) provided a model for handling the flexible job shop scheduling problem (FJSP), which was built using meta-heuristic algorithms like tabu search, GA, and ant colony optimization to reduce the time spent in planning and scheduling operations on the available set of machines. They have proven the value of the GA method in solving the FJSP problem, which yields positive outcomes when tested. Branco et al. (2016) focused the objective of minimizing completion time by using a hybrid genetic algorithm using heuristic rules in it. The metaheuristic techniques might improve the results of the mathematical models and proposed heuristics in a significant way (Hatami et al., 2015). This motivates our research to combine GA with a heuristic technique to reduce ET costs.

The paper is further divided into the sections listed below. Section two discusses the ET cost estimation for unequal earliness and tardiness penalties, and ET cost estimation for different earliness and tardiness penalties of jobs with a basic and combinatorial algorithm. Section three compares test results to a benchmark dataset and discusses those findings. Finally, the conclusions and effectiveness of the proposed algorithm are highlighted in section four.

2. Approaches for cost estimation with earliness and tardiness

In scheduling, earliness is a measure of finishing operations before due time. The period by which a job is early is known as the number of early days. Every early day costs a certain price which is known as earliness cost. Tardiness is a measure of a delay in executing certain operations. The period by which a job is late is known as the number of tardy days. Every tardy day costs a certain price which is known as tardiness cost. The JIT production concept supports the idea that being early as well as being late should be discouraged. Table 1 shows the parameters and variables used in the pseudo of algorithms.

Table 1. Description of parameters and variables.

Parameters	Description
j=1, 2, 3,…, N	Number of jobs
P_j	Processing time of job j
dd	The due date for jobs
df	Data_frame (j, P_j, *dd*) describes jobs with processing time and due date
α (*earlycost*)	Earliness cost of jobs
β (*tardycost*)	Tardiness cost of jobs
E_j	Earliness of job j
T_j	Tardiness of job j
α_j	Early cost of job j
β_j	Tardy cost of job j
Variables	**Description**
$\mid X \mid$	The number of jobs in list X which is empty at the beginning
$\mid Y \mid$	The number of jobs in list Y which is empty at the beginning
Δ	A factor that determines whether a scheduling problem is restricted or unrestricted
dt	Delay in the processing time of jobs at the start
δ	Due date as a decision variable
L_{amt}	The amount of time prior to the due date
R_{amt}	The amount of time left over after the due date
fc	Forward counter
bc	Backward counter
R_{eval}	The ratio value of early jobs that is P_j to α_j
R_{tval}	The ratio value of tardy jobs that is P_j to β_j
R_{e}	List of ratio value of jobs that finishes before (early) due date
R_{t}	List of ratio value of jobs that finishes after (tardy) due date

The fundamental goal function for the ET cost of N jobs is as follows, presuming the cost function is linear.

$$Cost = \sum_{j=1}^{N} \alpha_j E_j + \beta_j T_j \tag{1}$$

The Calculate Earliness and Tardiness cost (CETC) algorithm provides information on the cost of a sequence. Applying the cost of each job that is completed ahead of schedule results in the overall early cost of all jobs. Applying the cost of each job that is finished after the due date yields the total cost of late jobs. Later, the sum of these two costs is used to calculate the overall cost.

Algorithm CETC (Sequence, df, dd, earlycost, tardycost)

```
for $i$ in range ($|X|$)
        $cost_{earlyjobs} = cost_{earlyjobs} + (dd - P_j) \times earlycost$
for $i$ in range ($|Y|$)
        $cost_{tardyjobs} = cost_{tardyjobs} + (P_j - dd) \times tardycost$
$Totalcost = cost_{earlyjobs} + cost_{tardyjobs}$
return $Totalcost$
```

A schedule should ideally be designed such that *dd* is halfway through the jobs. It is impossible to get enough jobs completed before *dd* if the due date is very tight. This turns into a restricted version. If *dd* is not very constricted it turns out to be an unrestricted version. To achieve the goal of minimizing cost and maximizing profit, a procedure has been described to find the near-optimum sequence in which the jobs are processed. The problem can be categorized into three main groups based on ET cost.

2.1. Cost estimation with equal earliness and tardiness

In the ET problem, let us consider $\propto_j = 1, \beta_j = 1$ and common due date, the objective function can be given as:

$$Cost = \sum_{j=1}^{N} E_j + T_j \qquad (2)$$

Bari and Karande, (2022a) studied the cost of equal earliness and tardiness. To make their point obvious, they also provided numerical examples.

2.2. Cost estimation with Unequal Earliness and Tardiness (UET)

The problems with a common earliness cost but a different yet common tardiness cost are labelled under this section. A regular problem with the conditions: $\alpha_j = \alpha$ and $\beta_j = \beta$ follows the cost function defined as:

$$Cost = \sum_{j=1}^{N} \alpha\, E_j + \beta\, T_j \qquad (3)$$

Algorithm UET( ) provides a detail description of the cost estimation process with unequal earliness and tardiness. The algorithm is analogous to the equal earliness and tardiness algorithm. Initially assume two empty sets. The jobs are organized by processing time, starting with those that take the longest. Determine two values one by considering the value of being early (*V1*), and the other by considering the value of being late (*V2*). The formulation of values is shown in algorithm UET( ). If value *V1* is less than *V2* then insert the job in one set else insert the job in another set. Repeat this procedure until all jobs are scheduled. Now arrange jobs that complete before *dd* of one set in the longest processing time first rule and other sets' jobs that complete after *dd* in the shortest processing time first rule. Calculate the overall processing time of jobs that completes before the due date (Δ) that are jobs in one set. Now compare this value with the client's due date. If the client's due date is greater than Δ, it can be classified as an unrestricted scheduling problem otherwise restricted scheduling problem.

In an unrestricted solution, the manufacturer can ask the client to collect the finished jobs before the due date. If the client does not agree to collect the jobs early then, the manufacturer can add a buffer of certain no-working or idle days in the beginning. Otherwise, the task begins on the very first day. When there is a schedule conflict, the manufacturer begins working on the job from the first day.

Algorithm UET(df)

```
Input: $df(j, P_j, dd), \alpha, \beta$
Create two empty sets $X = \{\ \}$ and $Y = \{\ \}$
$df$ = sort (by [$P_j$], ascending = False)
$|X|$ = Number of jobs in $X$
$|Y|$ = Number of jobs in $Y$
for $i$ in range $N$
begin
    $V1 = \alpha \times |X|$
    $V2 = \beta \times (1 + |Y|)$
        if ($V1 < V2$)
            $X = df(j[i])$
            $|X| = |X| + 1$
        else ($V1 > V2$ or $V1 = V2$)
            $Y = df(j[i])$
            $|Y| = |Y| + 1$
end
$X$ = sort (by [$P_j$], ascending = False)
$Y$ = sort (by [$P_j$], ascending = True)
$\Delta = \sum_{j=1}^{N} [P_j]$ of jobs in set $X$
if $\Delta \le dd$ then
```

```
// The unrestricted scheduling case
$dt = dd - \sum_{j=1}^{|X|} [P_j]$ of jobs in set $X$
Process the jobs with a $dt$ time unit delay
$Sequence$ = merge ($X$, $Y$)
CETC ($Sequence, df, dd, \alpha, \beta$)
else
// The restricted scheduling case
    $L_{amt} = dd$
    $R_{amt} = (\sum_{j=1}^{|Y|} P_j) - dd$
    for $i$ in range of $N$
        if ($\alpha \times L_{amt} > \beta \times R_{amt}$)
        $Sequence[fc]$ = append($df(j[i])$)
        $P_j[fc] = df(P_j[i])$
        $L_{amt} = L_{amt} - df(P_j[i])$
        $fc = fc + 1$
        else
        $Sequence[bc]$ = append($df(j[i])$)
        $P_j[bc] = df(P_j[i])$
        $R_{amt} = R_{amt} - df(P_j[i])$
        $bc = bc - 1$
//Special case
if ($|X| > (N \times \beta)/(\alpha + \beta)$)
    $dt = dd - \sum_{j=1}^{N} [P_j]$ of jobs in set $X$
    Process the jobs with a $dt$ time unit delay
CETC ($Sequence, df, \delta, \alpha_j, \beta_j$)
```

The V-shaped sequence of jobs on a single machine is said to be the optimal schedule for ET problems (Kanet, 1981). To build a V-shaped schedule, the process begins on the first day which means the completion time is the summation of the processing time of all jobs. At each stage of the process, calculate the period available before the due date, and the period available after the due date. Formulation of the time at each stage is mentioned in algorithm UET( ). Compare the amount of time calculated before and after the due date. If the amount of time before the due date is greater than the time after the due date, arrange the job in the first vacant position in the sequence, else arrange the job in the last existing place in the sequence. Update the value of time before and after the due date after setting the job in its right location. This procedure will be carried out for each job in the system to obtain the optimum sequence.

A special sequence can be observed, wherein postponing the schedule's start can save overall costs. To find such a schedule, the obtained solution with the UET algorithm finally can be verified by ensuring the number of jobs that completes before the due date. Job processing is delayed by the variance between the due date and the total processing time of earlier jobs if the number of jobs that are finished ahead of schedule is larger than $[N\beta/(\alpha+\beta)]$. Finally, calculate the cost of an optimum sequence using the algorithm CETC( ).

2.3. Cost estimation with Job-Dependent Earliness and Tardiness (JDET)

In a manufacturing organization, it is typical practice to charge different rates for being early and late for different jobs. A regular problem with the conditions, α_j and β_j follows the cost function defined in Equation 1. Here due date is the decision variable. Calculate the due date (δ) as mentioned in JDET( ) algorithm. Divide the jobs into two groups, with the jobs that are finished on or before the due date being in one set and the jobs which are finished ahead of the due date being in the other. Jobs that begin later can be sequenced in the rising order of the ratio P_j/β_j, and jobs that finish earlier or on time can be sequenced in the declining order of the ratio P_j/α_j. Finally, calculate the cost of the obtained optimum sequence using the algorithm CETC( ) and due date (δ).

Algorithm JDET(df)

```
Input: $df(j, P_j, \alpha_j, \beta_j)$
Create two empty sets $X = \{\ \}$ and $Y = \{\ \}$
$df$ = sort (by $[P_j]$, ascending = False)
$|X|$ = Number of jobs in $X$
$|Y|$ = Number of jobs in $Y$
for $i$ in range $j$
begin
    $V1 = \alpha \times |X|$
    $V2 = \beta \times (1+|Y|)$
    if ($V1 < V2$)
        $X = df(j[i])$
        $|X| = |X| + 1$
    else ($V1 > V2$ or $V1 = V2$)
        $Y = df(j[i])$
        $|Y| = |Y| + 1$
end
$X$ = sort (by $[P_j]$, ascending = False)
$Y$ = sort (by $[P_j]$, ascending = True)
$\delta = \sum_{j=1}^{N} [P_j]$ of jobs in set $X$
Create two empty list for storing ratio $R_e = [\ ]$ and $R_t = [\ ]$
for $j$ in range $|X|$
    $R_{eval} = P_j / \alpha_j$
    $R_e$.append($R_{eval}$)
    $Sequence\_early$ =
    sort ($R_e$, ascending = False)
```

```
for $j$ in range $|Y|$
    $R_{tval} = P_j / \beta_j$
    $R_t$.append($R_{tval}$)
    Sequence_tardy = sort ($R_t$, ascending = True)
    Sequence = merge (Sequence_early, Sequence_tardy)
CETC (Sequence, df, $\delta$, $\alpha_j$, $\beta_j$)
```

2.4. Cost estimation with hybrid GA

The two combinatorial techniques used to calculate the overall ET cost are explained in this section. To reduce costs, the UET and JDET algorithms described in sections 2.2 and 2.3 respectively are combined with GA. Algorithms UET-GA( ) and JDET-GA( ) illustrate the steps involved in these approaches. In GA, the population is randomly initialized at the start. The sequence generated in UET is injected in UET-GA along with a randomly initialized population. Here the sequences of jobs are termed chromosomes. Now the population sequences are used, and the fitness function is applied to them. To imitate new sequences, the sequence is exposed to the crossover and mutation operators of GA. The two-point crossover is used to shuffle the positions of jobs. Using the crossover operator, offspring sequences are created from parent sequences. The crossover operator swaps the sub-sequences prior to and after randomly chosen positions between two sequences. Two new child sequences are created as an effect of crossing over two-parent sequences. After this, mutation is applied by sequentially swapping two arbitrarily chosen jobs. The ET cost function is used to evaluate the generated sequences, and the sequence with the lowest ET cost is deemed to be the optimal sequence for the iteration. The optimal sequence is now incorporated into the population of the following iteration as one of the members. This procedure is repeated for a given number of iterations. Finally, the UET-GA algorithm determines the optimal sequence with the least amount of ET cost. JDET-GA aids like UET-GA in optimizing an objective function. JDET-GA is a method that combines JDET and GA. Here the sequence generated with JDET is injected as one of the members of the population. To determine the ET cost of a sequence, JDET-GA computes the *dd* and provides it as a decision variable. The subsequent steps are the same as in UET-GA for finding the best sequence with the lowest cost.

Algorithm UET-GA(df) and JDET-GA(df)

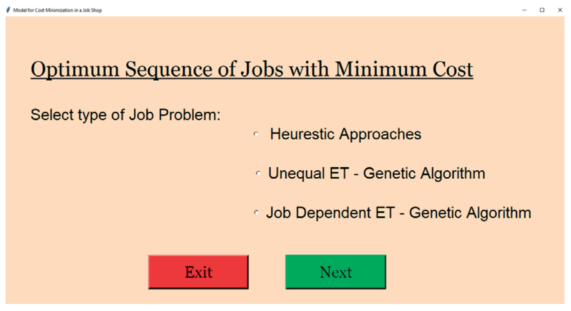
```
Input: df (j, $P_j$, $\alpha_j$, $\beta_j$)
if $\alpha_j=\alpha$ and $\beta_j=\beta$
    Best-sequence = sequence produced by UET ( )
else
Best-sequence=sequence produced by JDET( )
for iteration =1 to number of iterations
    Population = Randomly select 200 sequences
    Population = Population.Append(Best-sequence)
    Apply genetic operator crossover and mutation operation to generate offspring population
    Calculate cost CETC (Sequence, df, $\delta$, $\alpha_j$, $\beta_j$) and compare it
Best-sequence = Select sequence with less cost
Totalcost = CETC (Best-sequence, df, $\delta$, $\alpha_j$, $\beta_j$)
Print Best-sequence, Totalcost
```

3. Results and testing

The main goal of this article is to build a model to determine the optimal sequence of jobs for reducing ET cost using hybrid GA which includes UET-GA and JDET-GA. The Intel(R) Core(TM) i3-9100F CPU @ 3.60GHz computer system. Figure 1 shows a graphical user interface (GUI) of the model, developed using the Python programming language. The main window of the model is depicted in Figure 1(a), where the user can choose between the heuristic techniques, UET-GA and JDET-GA. If the user opts for heuristic approaches, one can pick between the UET and the JDET shown in Figure 1(b). Similar to this the user can choose UET-GA or JDET-GA. Figure 1(c) shows the GUI of the JDET-GA technique. The dataset is selected by clicking the “Browse” button and the number of iterations is entered as the stopping condition. Clicking the “Next” button displays a solution to the problem shown in Figure 1(d).

3.1. UET and UET-GA

This section displays the results of 10 instances with 10 jobs, each with the names D_UET_1 to D_UET_10, here D_ represents the data set and the number represents the serial number of instances with unequal ET cost. For UET and UET-GA algorithms, a data set with the job’s processing time, common due date, same earliness costs for all jobs, and tardiness costs for all jobs is required. The processing times for instances and factor *h* are taken from the benchmark dataset of Biskup and Feldman (2001).

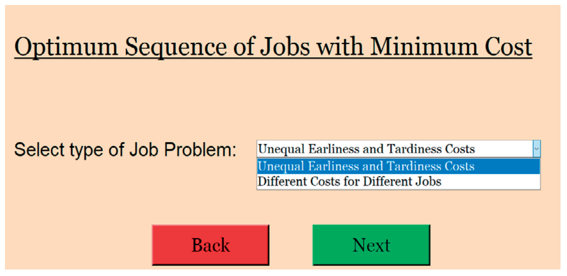


Optimum Sequence of Jobs with Minimum Cost

Select type of Job Problem:

- Heurestic Approaches
- Unequal ET - Genetic Algorithm
- Job Dependent ET - Genetic Algorithm

(a)

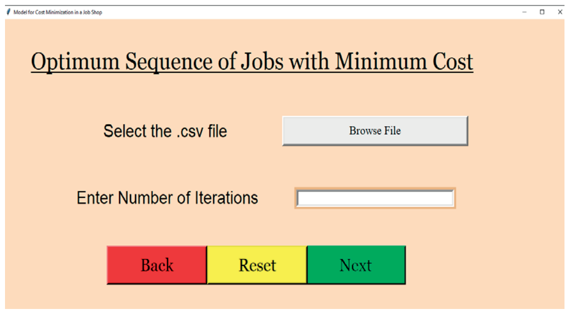


Optimum Sequence of Jobs with Minimum Cost

Select type of Job Problem: Unequal Earliness and Tardiness Costs

- Unequal Earliness and Tardiness Costs
- Different Costs for Different Jobs

(b)

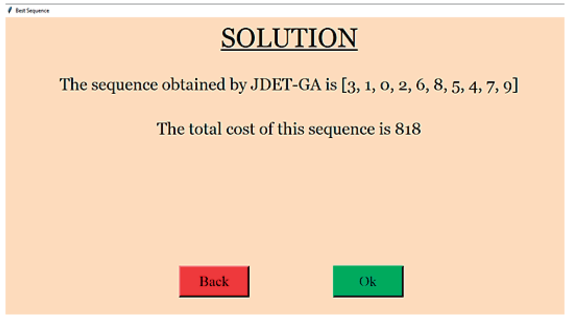


Optimum Sequence of Jobs with Minimum Cost

Select the .csv file

Enter Number of Iterations

(c)



SOLUTION

The sequence obtained by JDET-GA is [3, 1, 0, 2, 6, 8, 5, 4, 7, 9]

The total cost of this sequence is 818

(d)

Figure 1. GUI of model.

3.1.1. Unrestricted scheduling problem

An unrestricted scheduling case that is discussed in section 2.2 is solved by using the UET algorithm. For each instance, a fictitious cost for being early or late is used. Since the scheduling problem is unrestricted, the due date is established with h = 0.8 and it is calculated as $dd=Sum(P_j)\times h$. Table 2 presents the results using the UET technique with input, earliness and tardiness costs for each data set. The sequence can be observed, where the overall cost can be minimized by introducing idle time at the start of the schedule. The algorithm finds the delay and then finds the cost of the schedule.

Table 2. Results of UET (Unrestricted, N = 10).

Data Set	Sum(P)	dd	α	β	Delay Processing (dt)	Total Cost
D_UET_1	116	93	4	5	20	1037
D_UET_2	129	103	10	2	72	804
D_UET_3	125	100	6	12	12	1932
D_UET_4	102	82	7	15	3	1773
D_UET_5	94	75	8	4	38	844
D_UET_6	88	70	4	2	31	446
D_UET_7	103	82	9	10	17	1727
D_UET_8	79	63	9	1	45	205
D_UET_9	92	74	1	2	4	195
D_UET_10	127	102	5	7	19	1324

When UET-GA is used on the same data set for 100, 200, 300, 400, 500, and 1000 iterations, it is observed that the cost is increased when compared with the results of UET as shown in Table 2. The cost for each iteration and the minimum cost among all iterations for UET-GA is shown in Table 3. It is advisable to include idle time, which delays the initial processing of jobs and reduces the cost value. Therefore, if the schedule is unrestricted, the UET technique performs better than UET-GA but considering the delay by dt unit of time in processing the sequence.

Table 3. Results of UET-GA (Unrestricted, N = 10).

	No. of Iterations						Minimum
Data Set	100	200	300	400	500	1000	Cost
D_UET_1	1174	1174	1174	1174	1174	1174	1174
D_UET_2	2512	2512	2502	2512	2522	2504	2502
D_UET_3	1956	1956	1956	1956	1950	1956	1950
D_UET_4	1785	1785	1785	1785	1785	1785	1785
D_UET_5	1284	1288	1284	1284	1284	1280	1280
D_UET_6	722	718	720	722	724	722	718
D_UET_7	1965	1965	1960	1965	1965	1965	1960
D_UET_8	905	906	905	905	905	905	905
D_UET_9	197	197	196	196	196	195	195
D_UET_10	1430	1435	1430	1430	1430	1430	1430

3.1.2. Restricted scheduling problem

The data set has a more restricted due date, and it is calculated with h = 0.2 as dd = $Sum(P_j)\times h$. The UET algorithm described in section 2.2 is used to find the value of the total cost for the data set. The

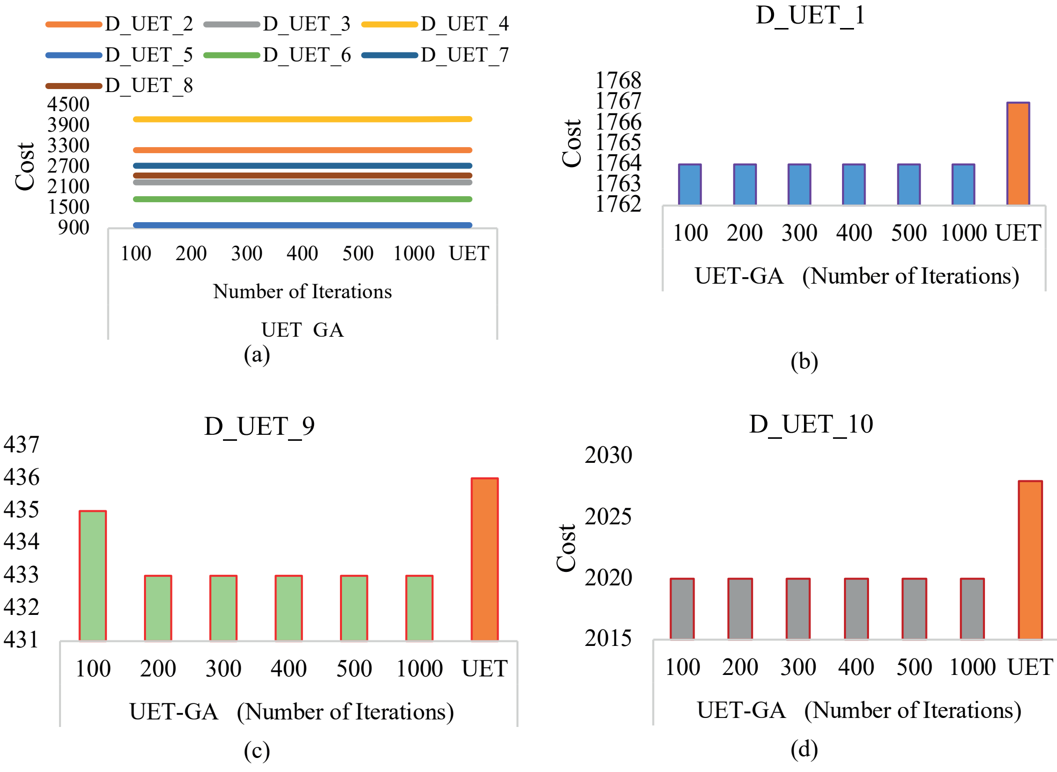


Figure 2. Trends of the objective function using UET and UET-GA (N = 10).

identical data set was processed using the UET-GA technique described in section 2.4. The value of the objective function was computed using several iterations, including 100, 200, 300, 400, 500, and 1000. Results of these 10 instances with 10 jobs using UET and UET-GA with various iterations are shown in Table 4.

Table 4. Results of UET (Restricted) and UET-GA (N = 10).

	Number of Iterations						
Data Set	100	200	300	400	500	1000	UET
D_UET_1	1764	1764	1764	1764	1764	1764	1767
D_UET_2	3196	3196	3196	3196	3196	3196	3196
D_UET_3	2250	2250	2250	2250	2250	2250	2250
D_UET_4	4102	4102	4102	4102	4102	4102	4109
D_UET_5	992	992	992	992	992	992	992
D_UET_6	1755	1755	1755	1755	1755	1755	1755
D_UET_7	2737	2737	2737	2737	2737	2737	2737
D_UET_8	2450	2450	2450	2450	2450	2450	2450
D_UET_9	435	433	433	433	433	433	436
D_UET_10	2020	2020	2020	2020	2020	2020	2028

Figure 2 (a-d) visually displays cost trends for UET and UET-GA for 10 instances with 10 jobs. It has been found that 6 instances produce the same outcome. For the other 4 instances, the costs for UET and UET-GA differ. In UET, it takes a fraction of a second to find the cost, but not in UET-GA. As indicated in Figure 3, it takes 6 seconds to 60 seconds for 100 to 1000 iterations. Additionally, instances of 20 jobs are tested and the results are shown in Table 5. It has been observed that 4 out of 10 instances (instances are highlighted in bold) have obtained less cost after applying UET-GA as compared to UET while the remaining 6 have the same objective function value. Results and trends of the cost function for UET and UET-GA are shown in Figure 4 (a-d). Therefore, if there are more jobs in the dataset, it will affect the cost and UET-GA will perform better than UET even if it would take a little longer time to give results.

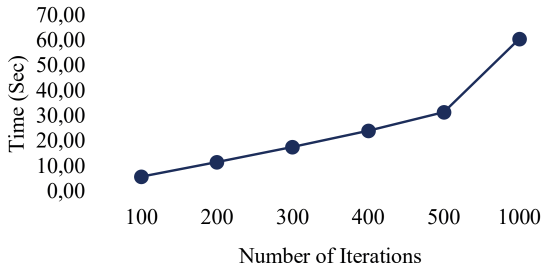


Figure 3. Time required to find the value of an objective function using UET-GA.

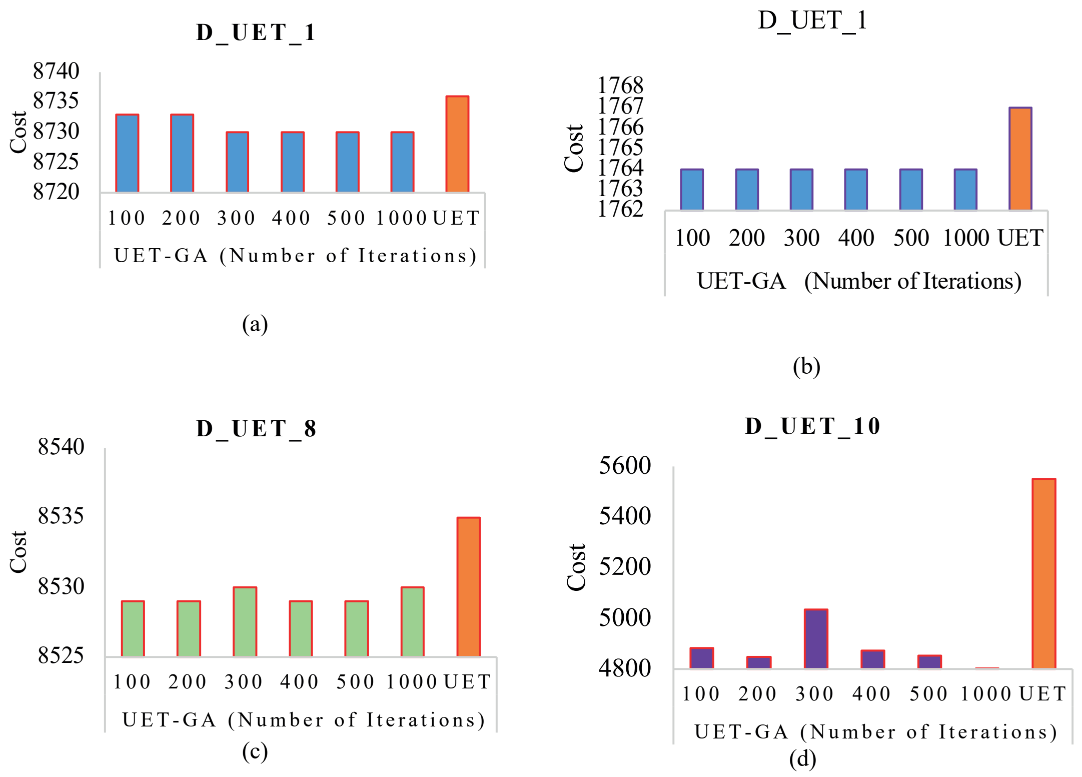


Figure 4. Trends of the objective function using UET and UET-GA (N = 20).

3.2. JDET and JDET-GA

In this section, the results of JDET and JDET-GA are compared and discussed. A set of 280 test instances created by Biskup and Feldman (2001) and made accessible online at (http://people.brunel.ac.uk/mastjjb/jeb/orlib/schinfo.html) were used in the numerical testing. The instance set is distributed into seven sets of N = 10, 20, 50, 100, 200, 500, and 1000 jobs each, with ten instances in every group. The problem is categorized as more restricted or less restricted against the common due date based on the values of h = 0.2, 0.4, 0.6, and 0.8. The 10 instances with 10 jobs from 280 test problems are tested with the proposed algorithm. These instances are named JDET_1 to JDET_10. JDET-GA conducts a global search for the schedule. The suggested JDET obtains cost values in a fraction of a second, however using JDET-GA, it takes about 60 seconds for 1000 iterations to acquire costs for 10 jobs. Despite taking more time, it offers the best value for the objective function. A due date that was calculated and denoted as a decision variable using the JDET approach is shown in Table 6. By using the formula $dd = Sum(P_j) \times h$, Biskup and Feldman (2001) estimated the due date which is also recorded in Table 6.

Table 7 gives the best objective function results for the 10 instances produced using the two algorithms. The last column of Table 7 represents the optimal value of each instance recorded by Biskup and Feldman (2001).

Table 5. Results of UET (Restricted) and UET-GA (N = 20).

	Number of Iterations						
Data Set	100	200	300	400	500	1000	UET
D_UET_1	8733	8730	8730	8730	8730	8730	8736
D_UET_2	9992	9992	9992	9992	9992	9992	9992
D_UET_3	1446	1446	1446	1446	1446	1446	1446
D_UET_4	4762	4762	4762	4762	4762	4762	4762
D_UET_5	17285	17285	17285	17285	17285	17285	17285
D_UET_6	2446	2442	2442	2446	2448	2442	2452
D_UET_7	3864	3864	3864	3864	3864	3864	3864
D_UET_8	8529	8529	8530	8529	8529	8530	8535
D_UET_9	4762	4762	4762	4762	4762	4762	4762
D_UET_10	4884	4848	5034	4802	4853	4873	5551

Table 6. Given *dd* and *dd* as Decision Variable.

		Due Date				
N = 10	Sum(P_j)	*h*=0.2	*h*=0.4	*h*=0.6	*h*=0.8	Decision Variable
JDET_1	116	23	46	70	93	76
JDET _2	129	26	52	77	103	77
JDET _3	125	25	50	75	100	87
JDET _4	102	20	41	61	82	59
JDET _5	94	19	38	56	75	51
JDET _6	88	18	35	53	70	63
JDET _7	103	21	41	62	82	68
JDET _8	79	16	32	47	63	52
JDET _9	92	18	37	55	74	68
JDET _10	127	25	51	76	102	93

Table 7. Comparison of cost computed with Heuristic JDET, JDET-GA and *h* = 0.8.

Due Date	Decision Variable		Sum(P_j)* 0.8
N = 10	Heuristic JDET	JDET-GA	*h* = 0.8
JDET _1	893	818	818
JDET _2	932	877	615
JDET _3	982	977	793
JDET _4	896	815	803
JDET _5	690	521	521
JDET _6	925	849	755
JDET _7	1521	1147	1083
JDET _8	616	599	540
JDET _9	624	562	554
JDET _10	788	695	671

Figure 5 displays an objective function's cost for ten instances graphically. It was found that JDET-GA outperformed JDET in terms of performance. Although it is calculated for a due date with *h* = 0.8, this due date is less restrictive, the cost estimations for 10 instances taken from the benchmark dataset are less than JDET-GA. This means that the cost will increase if the due date is more restrictive. The proposed algorithm gives near-about cost with a suggested due date lesser than the less restrictive due date.

Table 8 displays the JDET and JDET-GA fitness values for *N* = 10, 20, 50, 100, 200, 500, and 1000 jobs for 5000 iterations. These findings are based on a benchmark dataset (First instance) with 10, 20, 50, 100, 200, 500, and 1000 jobs each. The optimal value recorded in the benchmark dataset for the instances is 818 (*N* = 10), 2986 (*N* = 20), 17990 (*N*=50), 72019 (*N* = 100), 254268 (*N* = 200), 1581233 (*N* = 500), 6411581(*N* = 1000). When put up against JDET, JDET-GA performs better, if the cost is calculated using JDET, it is shown that the cost increases in all testing instances. The benchmark dataset's optimal value is lower than JDET-GA, but the *dd* is less restrictive as compared to the *dd* as a decision variable. When compared to the optimal value for the benchmark dataset, JDET-GA gives approximately the same cost. For situations with 10 to 500 jobs, the difference in the cost increases in the range of 0% to 8% except for 1000 jobs. Thus JDET-GA is used to determine the optimal sequence and suggest a *dd* that reduces the ET cost.

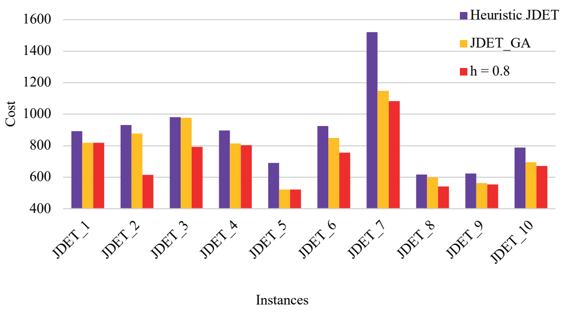


Figure 5. Objective function values for JDET, JDET-GA and DD with h=0.8.

Table 8. Comparison of fitness value for JDET, JDET-GA and benchmark dataset.

N	Sum(P_j)	*dd* = 0.8× Sum(P_j)	Optimal value recorded in benchmark dataset	*dd* - Decision variable	Fitness Value JDET-GA	Fitness Value JDET
10	116	93	818	76	818	893
20	217	174	2986	121	3080	5253
50	549	439	17 990	338	19 401	29 066
100	1136	909	72 019	650	75 101	118 560
200	2129	1703	254 268	1173	274 268	467 681
500	5217	4174	1 581 233	2903	1 688 888	2 688 036
1000	10 611	8489	6 411 581	6091	8 449 072	11 445 102

4. Conclusions

In this paper, a novel UET-GA and JDET-GA are introduced. The results show that the proposed UET-GA and JDET-GA perform better in a majority of instances when used to solve the single-machine scheduling problem with ET costs. To improve the quality of the sequence to reduce the cost and time necessary to converge to optimal outcomes, a population initialization process is introduced by adding a sequence obtained with heuristic UET and JDET approaches. In the unrestricted scheduling problem, the UET heuristic results in lower ET costs than UET-GA, but it also introduces idle time in the beginning, increasing the completion time. In the restricted scheduling problem, UET-GA performs better than UET, especially if there are more jobs in the dataset. JDET-GA is employed to find the optimal sequence and suggests a due date. It lowers the cost than JDET heuristic approach in all the instances. The suggested due date is tighter than the due date ($h = 0.8$) reflected in the benchmark dataset and significantly lowers the cost.

A probable direction for further research includes using the same UET-GA and JDET-GA methodology in the situation of multi-machine scheduling problems with general ET penalties to find an optimum solution in case of real test problems. This research can be further extended with a distinct due date for each job.

References

Akande, S., & Ajisegiri, G. (2022). Three classes of Earliness-Tardiness (E/T) scheduling problems. *Fuel Communications, 10*, 100047. https://doi.org/10.1016/j.jfueco.2021.100047

Alidaee, B., Li, H., Wang, H., & Womer, K. (2021). Integer programming formulations in sequencing with total earliness and tardiness penalties, arbitrary due dates, and no idle time: A concise review and extension. *Omega (United Kingdom), 103*, 102446. https://doi.org/10.1016/j.omega.2021.102446

Allaoua, H., & Osmane, I. (2010). Variable parameters lengths genetic algorithm for minimizing earliness-tardiness penalties of single machine scheduling with a common due date. *Electronic Notes in Discrete Mathematics, 36*(C), 471–478. https://doi.org/10.1016/j.endm.2010.05.060

Arık, O. A. (2020). Comparisons of metaheuristic algorithms for unrelated parallel machine weighted earliness/tardiness scheduling problems. *Evolutionary Intelligence, 13*(3), 415–425. https://doi.org/10.1007/s12065-019-00305-7

Arık, O. A., Schutten, M., & Topan, E. (2022). Weighted earliness/tardiness parallel machine scheduling problem with a common due date. *Expert Systems with Applications, 187*. https://doi.org/10.1016/j.eswa.2021.115916

Baker, K. R., & Scudder, G. D. (1990). Sequencing with earliness and tardiness penalties. A review. *Operations Research, 38*(1), 22–36. https://doi.org/10.1287/opre.38.1.22

Bari, P., & Karande, P. (2022a). Cost Minimization in a Scheduling Problem with Unrestricted and Restricted Common Due Date. *Recent Advances in Industrial Production. Lecture Notes in Mechanical Engineering. Springer, Singapore*, 269–278. https://doi.org/10.1007/978-981-16-5281-3_25

Bari, P., Karande, P., & Menezes, J. (2022b). Simulation of Job Sequencing for Stochastic Scheduling With a Genetic Algorithm. *Operational Research in Engineering Sciences: Theory and Applications, 5*(3), 17–39. https://doi.org/10.31181/oresta060722075b

Beyranvand, M. S., Peyghami, M. R., & Ghatee, M. (2012). On the quadratic model for unrelated parallel machine scheduling problem with restrictive common due date. *Optimization Letters, 6*(8), 1897–1911. https://doi.org/10.1007/s11590-011-0385-0

Biskup, D., & Feldmann, M. (2001). Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. *Computers and Operations Research, 28*(8), 787–801. https://doi.org/10.1016/S0305-0548(00)00008-3

Branco, R. M., Coelho, A. S., & Mayerle, S. F. (2016). Hybrid genetic algorithms: solutions in realistic dynamic and setup dependent job-shop scheduling problems. *International Journal of Production Management and Engineering, 4*(2), 75. https://doi.org/10.4995/ijpme.2016.5780

Carlier, J., Hermès, F., Moukrim, A., & Ghédira, K. (2010). Exact resolution of the one-machine sequencing problem with no machine idle time. *Computers and Industrial Engineering, 59*(2), 193–199. https://doi.org/10.1016/j.cie.2010.03.007

Chen, Y. W., Lu, Y. Z., Ge, M., Yang, G. K., & Pan, C. C. (2012). Development of hybrid evolutionary algorithms for production scheduling of hot strip mill. *Computers and Operations Research, 39*(2), 339–349. https://doi.org/10.1016/j.cor.2011.04.009

Gupta, A., & Kumar, S. (2016). Flow shop scheduling decisions through Techniques for Order Preference by Similarity to an Ideal Solution (TOPSIS). *International Journal of Production Management and Engineering, 4*(2), 43. https://doi.org/10.4995/ijpme.2016.4102

Hatami, S., Ruiz García, R., & Romano, C. A. (2015). The Distributed Assembly Parallel Machine Scheduling Problem with eligibility constraints. *International Journal of Production Management and Engineering, 3*(1), 13. https://doi.org/10.4995/ijpme.2015.3345

Irani, S., & Pruhs, K. (2005). Algorithmic problems in power management, ACM SIGACT News*, 36*(2), 63–76. https://doi.org/10.1145/1067309.1067324

Kanet, J. J. (1981). Minimizing the average deviation of job completion times about a common due date. *Naval Research Logistics, 28*(4), 643–651. https://doi.org/10.1002/nav.3800280411

Kanet, J. J., & Sridharan, V. (2000). Scheduling with inserted idle time: problem taxonomy and literature review. *Operations Research, 48*(1), 99–110. https://doi.org/10.1287/opre.48.1.99.12447

Khoshjahan, Y., Najafi, A. A., & Afshar-Nadjafi, B. (2013). Resource constrained project scheduling problem with discounted earliness-tardiness penalties: Mathematical modeling and solving procedure. *Computers and Industrial Engineering, 66*(2), 293–300. https://doi.org/10.1016/j.cie.2013.06.017

Liaw, C. F. (1999). A branch-and-bound algorithm for the single machine earliness and tardiness scheduling problem. *Computers and Operations Research, 26*(7), 679–693. https://doi.org/10.1016/S0305-0548(98)00081-1

M'Hallah, R. (2007). Minimizing total earliness and tardiness on a single machine using a hybrid heuristic. *Computers and Operations Research, 34*(10), 3126–3142. https://doi.org/10.1016/j.cor.2005.11.021

Ow, P. S., & Morton, T. E. (1989). The single machine early/tardy problem. *Management Science, 35*(2), 177–191. http://www.jstor.org/stable/2631910

Oyetunji, E. O. (2009). Some common performance measures in scheduling problems: Review article. *Research Journal of Applied Sciences, Engineering and Technology, 1*(2), 6–9.

Plateau, M. C., & Rios-Solis, Y. A. (2010). Optimal solutions for unrelated parallel machines scheduling problems using convex quadratic reformulations. *European Journal of Operational Research, 201*(3), 729–736. https://doi.org/10.1016/j.ejor.2009.03.049

Raghavan, V. A., Yoon, S. W., & Srihari, K. (2018). A modified Genetic Algorithm approach to minimize total weighted tardiness with stochastic rework and reprocessing times. *Computers and Industrial Engineering, 123*, 42–53. https://doi.org/10.1016/j.cie.2018.06.002

Rolim, G. A., & Nagano, M. S. (2020). Structural properties and algorithms for earliness and tardiness scheduling against common due dates and windows: A review. *Computers and Industrial Engineering, 149.* https://doi.org/10.1016/j.cie.2020.106803

Schaller, J., & Valente, J. M. S. (2013). A comparison of metaheuristic procedures to schedule jobs in a permutation flow shop to minimise total earliness and tardiness. *International Journal of Production Research, 51*(3), 772–779. https://doi.org/10.1080/00207543.2012.663945

Shabtay, D., & Steiner, G. (2007). A survey of scheduling with controllable processing times. *Discrete Applied Mathematics, 155*(13), 1643–1666. https://doi.org/10.1016/j.dam.2007.02.003

Stanković, A., Petrović, G., Ćojbašić, Ž., & Marković, D. (2020). An application of metaheuristic optimization algorithms for solving the flexible job-shop scheduling problem. *Operational Research in Engineering Sciences: Theory and Applications, 3*(3), 13–28. https://doi.org/10.31181/oresta20303013s

Wagner, B. J., Davis, D. J., & Kher, H. V. (2002). The production of several items in a single facility with linearly changing demand rates. *Decision Sciences, 33*(3), 317–346. https://doi.org/10.1111/j.1540-5915.2002.tb01647.x

Woodruff, D. L., & Spearman, M. L. (1992). Sequencing and batching for two classes of jobs with deadlines and setup times. *Production and Operations Management, 1*(1), 87–102. https://doi.org/10.1111/j.1937-5956.1992.tb00341.x

Yuce, B., Fruggiero, F., Packianather, M. S., Pham, D. T., Mastrocinque, E., Lambiase, A., & Fera, M. (2017). Hybrid Genetic Bees Algorithm applied to single machine scheduling with earliness and tardiness penalties. *Computers and Industrial Engineering, 113,* 842–858. https://doi.org/10.1016/j.cie.2017.07.018