SCITECH
RESEARCH ORGANISATION

Volume 14, Issue 1
Published online: September 07, 2018|

Journal of Progressive Research in Mathematics
www.scitecresearch.com/journals

Solving Machine Scheduling Problem under Fuzzy Processing Time using the Simulated Annealing Method

Al-zuwaini Mohammed Kadhim [1], Shaker K. Ali [2], Marwa Mohammed Kassim[3]
[1]University of Thi-Qar, mkzz50@yahoo.com
[2]University of Thi-Qar, shaker@utq.edu.iq
[3]University of Thi-Qar, mm8764270@gmail.com

Abstract

In this paper, we describe the problem of sequencing a set of n jobs on single machine was considered to minimize multiple objectives function (MOF). The objective is to find the approximate solutions for scheduling n independent jobs to minimize the objective function consists from asum of weighted number of earlyjobs and weighted number of tardy jobs with fuzzy processing time.This problem is denoted by: $(1/ /\sum_{j=1}^{n} N_{h_j E_j} + \sum_{j=1}^{n} N_{w_j T_j})$. To resolve it we proposed the Average High Ranking (AHR) method to obtain a processing time generated from fuzzy processing time, calculate the costs and reach to total penalty cost. Since our problem is Strongly NP-hard in normal form, we used Simulated Annealing. It solved the problem with up to 12000 jobs in 30 seconds.

Keywords: Scheduling; Single machine; Fuzzy processing time; Simulated Annealing Heuristics; Weighted number of early jobs; Weighted number of tardy jobs.

1. Introduction

In this paper, processing of fuzzy-time is regarded in three conditions that are ranging from best to worse conditions as follows; high or favorable one, medium or normal one, and finally thebad or worst one[1]. Meaning that the number of tardy criterion is considered to be a normal method for assessing performance with due dates, even though this criterion disregards penalties with jobs that are early completed. Baker and Scudder (1990) [2] has investigated the sequences that possesses penalties of both tardiness and earliness in an environment with JIT scheduling, the jobs which finish earlier are saved with completed decent record while waiting for their due date to end. On the other hand, jobs which finish late with the assigned due dates can initiate shutdown operation with the client. For that reason, optimal scheduling refers to any completed job with its allocated due date. This may be interpreted to a scheduling objective by numerous means.

The best apparent objective is to decrease deviations in job completion time (JCT) nearby these due dates with indeterministic time. In penalizing notion tardiness and earliness has both to be produced with novel and fast evolving stream of studies in the scheduling area of research. Since usage of penalties in tardiness and earliness with fuzzified environment promises an irregular measure of performance, it paved the way towards novel operational topics in the enterprise of solution procedures.

Scheduling of independent n jobs in a single machine will be regarded here to minimize bi-criteria. The aim lies in finding a schedule minimization to the summation of weighted number of early jobsin

addition to weighted number of tardy jobs under fuzzy processing time, in employing local search algorithm Simulated Annealing (SA). This problem is denoted by $(1/\!/\sum_{j=1}^{n} N_{h_j E_j} + \sum_{j=1}^{n} N_{w_j T_j})$

Where h_j, w_j are weighted of early job j, weighted of tardy job j respectively.

$$N_{E_j} = \begin{cases} 1 & if\ c_j < d_j \\ 0 & otherwise \end{cases} \quad (1)$$

$$N_{T_j} = \begin{cases} 1 & if\ c_j > d_j \\ 0 & otherwise \end{cases} \quad (2)$$

The problem in normal case (that is without fuzzy processing time) was considered by H. Soroush (2007)[3] and he show that it is NP-hard. We used simulated Annealing method to find approximate solutions, for our problem solved approximately with up to 12000 jobs.

2. Formulation of the problem

Single machine scheduling models seem to be very important for understanding and modeling multiple machines models. A set N={1,2,…,n} of n independent jobs has to be scheduledon a single machine in orderto minimize a given criterion.This study concerns the one machine scheduling problem with multiple objectives function which is denoted by $(1/\!/\sum_{j=1}^{n} N_{h_j E_j} + N_{w_j T_j})$.

In this problem, preemption is not allowed, no precedence relation among jobs is assumed and all jobs are available at the time zero, (that is r_j=0 $\forall j$). Each job j has fuzzy processing time a_j, b_j, c_j, due dated_j, weighted of early job j h_j and weighted of tardy job j w_j.

The start time of job j is denoted by t_j, where t_1=0 and $t_j = \sum_{i=1}^{j-1} p_i$ and its completion time by C_j, $(C_j = t_j + p_j)$, if job j completed befor it's due date $(C_j < d_j$ then N_{E_j}=1) and job j is said to be early otherwise ($C_j \geq d_j$, then N_{E_j}=0), and job j is said to be tardy job or just in time, for each job j can be calculate the slack time $Sl_j = |A_j - d_j|$. In this paper, next jobs are scheduled in increasing order of their slack time. Consider the processing time in fuzzy environment (a, b, c), which is integer numbers. We calculate the Average High Ranking (AHR) Method to generate a processing time from a fuzzy processing time.

$$AHR = [3b + (c - a)]/\,3 \quad (3)$$

Where a, b, cis fuzzy processing time.

For a given schedule we can calculate weighted number of early jobs$N_{h_j E_j}$, as follow:

$$N_{h_j E_j} = \begin{cases} h_j & if \quad d_j > C_j \\ 0 & if \quad d_j \leq C_j \end{cases} \quad (4)$$

And weighted number of tardy jobs $N_{w_j T_j}$:

$$N_{w_j T_j} = \begin{cases} w_j & if \quad C_j > d_j \\ 0 & if \quad C_j \leq d_j \end{cases} \quad (5)$$

The objective is to find the schedule $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$ of the jobs that minimize the total cost R which is formulated in mathematic form as:

$$
\left.
\begin{aligned}
&min\, R = min_{\pi\in\delta}\{\textstyle\sum_{j=1}^{n} N_{h_j E_j} + \sum_{j=1}^{n} N_{w_j T_j}\}\\
&\textbf{\textit{subject to}}\\
&\quad C_{\pi(j)} \geq p_{\pi(j)} \qquad\qquad j = 1,2,\dots,n\\
&\quad C_{\pi(j)} = C_{\pi(j-1)} + P_{\pi(j)}\ j = 2,3,\dots,n\\
&\quad N_{E_{\pi(j)}} \in \{0,1\} \qquad\qquad j = 1,2,\dots,n\\
&\quad N_{T_{\pi(j)}} \in \{0,1\} \qquad\qquad j = 1,2,\dots,n\\
&p_{\pi(j)} > 0,\ \ d_{\pi(j)} > 0,\ \ w_{\pi(j)} > 0,\ \ h_{\pi(j)} > 0
\end{aligned}
\right\} \dots\dots\dots\dots \textbf{(H)}
$$

Where δ the set of all feasible solutions, π(j)denoted the positionofjin the ordering π.

3. Algorithm

Step 1: Determine Average High Ranking (AHR) of the fuzzy processing time (a, b, c) of all the jobs.

Step 2: Determine the slack time of all the jobs$Sl_j = \left| A_j - d_j \right|$.

Step 3: Ordered the jobs in MST rule. If more than job have the same slack time then breaking ties the sequence by EDD rule.

Step 4: Using the sequence obtained in step 3 find thepenalty cost of all the jobs using weighted of early jobs (h_j) and weighted of tardy jobs (w_j).

Step 5:END.

Example 3.1

Consider four jobs having fuzzy processing time, single machine,due dates,weighted early jobs(h_j)and weighted tardy jobs (w_j) are given.

Table (1): The initial data for example 3.1

Job	p_j	d_j	h_j	w_j
1	1 2 4	5	2	3
2	38 9	11	4	6
3	5 7 8	25	7	9
4	7 8 10	30	10	12

Solution:

Table (2): The calculated Average High Ranking (AHR) and Slack time for example 3.1

Job	p_j	(AHR) = P_j	d_j	Sl_j	h_j	w_j
1	1 2 4	3	5	2	2	3
2	3 8 9	10	11	1	4	6
3	57 8	8	25	17	7	9
4	7 8 10	9	30	21	10	12

Then MST rule gives the sequence:

S = 2 — 1 — 3 — 4

The table (3) shows the flow time of the system and penalty cost.

Table (3): The calculated penalty cost for example 3.1

Job	P_j	d_j	c_j	h_j	w_j	cost
2	10	11	10	4	6	4
1	3	5	13	2	3	3
3	8	25	21	7	9	7
4	9	30	30	10	12	0

Penalty cost is = 14

4. Simulated Annealing (SA) Heuristics

The simulated annealing algorithm is a way of finding solutions to problems which have a large set of possible solutions, in an analogous fashion to the physical annealing of solids to attain minimum internal energy states. The basic idea is to create a path through the solution space, from one solution to another nearby solution, leading ultimately to the optimum solution. In generating this path, solutions are chosen from the locality of the preceding solution by a probabilistic function of the improvement gained by this move. So, stages are not strictly required to produce improved solutions, but each stage has a certain probability of leading to improvement; at the start, all stages all equally likely, but as the algorithm progresses, the tolerance for solutions worse than the current one decreases, eventually to the point where only improvements are accepted. In this way, the algorithm can attain the optimum solution without becoming trapped in local optima [4, 5].

This tells us that we can have some confidence in the answer Created by a well-formulated simulated annealing implementation, but says nothing about how long it might take to find that answer. It is often used when the search space is discrete (e.g., all tours that visit a given set of cities).

The method was independently described by Kirkpatrick, et.al., (1983)[6] and by V. Černý (1985)[7]. The method is an adaptation of theMetropolis-Hastings algorithm,a Monte Carlo method to generate sample states of a thermodynamic system, invented by M.N. Rosenbluth and published in a paper by Metropolise, et.al., (1953) [8].

The following structure gives the outline of (SA):

Algorithm:[5]

S_C Candidate schedule.

S_0 Best schedule found so far.

S_K Schedule constructed at K^{th} iteration (K = iteration counter)

$G(S_0)$ Best schedule value (Aspiration criterion).

$G(S_K)$ Value of schedule constructed at K^{th} iteration.

$G(S_C)$ Value of candidate schedule.

$P(S_K,S_C)$ Probability of moving from S_K schedule to S_C schedule at KIteration

$$P(S_K , S_C) = \exp\left(\frac{G(S_K)-G(S_C)}{\beta_K}\right)$$

Where,

β_K is called cooling parameter in annealing terminology usually,

$\beta_K = a^K$ where $a \in [0,1]$.

First Step: Initialization

Set K=1, set β_1 equal to given value.

From starting sequence by any heuristic; call it S_1

Let $S_0 = S_1$, then $G(S_0) = G(S_1)$

Second Step: Conditioning

Select S_C from S_K

IF $G(S_0) < G(S_C) < G(S_K)$,

Then $S_{K+1} = S_C$

Go to Third step

IF $G(S_C) < G(S_0)$,

Then $S_0 = S_{K+1} = S_C$

Go to Third step

IF $G(S_C) > G(S_k)$,

Then generate a random number $U_K \sim [0,1]$

IF $U_K \leq P(S_K , S_C)$,

Then $S_{K+1} = S_C$

ELSE $S_{K+1} = S_K$

Go to Third step

Third Step: Finalization

Set $\beta_{K+1} \leq \beta_K$ and Set K= K+1

IF $K \leq N$

Then Go to Second Step

ELSE Stop

Example 4.1

Solve the problem (1/ / $\sum_{j=1}^{n} N_{h_j E_j} + \sum_{j=1}^{n} N_{w_j T_j}$) using SA method. Apply the technique with four iteration (K ≤ 4), and N = 4.Use U values ={0.8, 0.01, 0.02, 0.43}, Take initial value of β = 0.9 , using the data of EX (3.1).

Solution

$\beta_1 = 0.9$, K = 1, $S_1 = \{j_2, j_1, j_3, j_4\}$

Set $S_0 = S_1$

$G(S_0) = 14$, $G(S_1) = 14$,

Let $S_C = \{j_1, j_2, j_3, j_4\}$,

$G(S_C) = 15$,

$G(S_C) > G(S_1)$,

$U_1 = 0.8$

$P(S_1, S_C) = exp\left\{\frac{G(S_1)-G(S_C)}{\beta_1}\right\} = 0.2865$

$U_1 > P(S_1, S_C)$

set $S_2 = S_1$, $S_2 = \{j_2, j_1, j_3, j_4\}$, $G(S_2) = 14$

$\beta_2 = (\beta_1)^2 = (0.9)^2 = 0.81$

K = 2, N = 4

Let $S_C = \{j_2, j_1, j_4, j_3\}$,

$G(S_C) = 26$,

$G(S_C) > G(S_2)$

$U_2 = 0.01$

$P(S_2, S_C) = exp\left\{\frac{G(S_2)-G(S_C)}{\beta_2}\right\} = 0.0000003681$

$U_2 > P(S_2, S_C)$

set $S_3 = S_2$, $S_3 = \{j_2, j_1, j_3, j_4\}$, $G(S_3) = 14$

$\beta_3 = (\beta_1)^3 = 0.729$

K = 3, N = 4,

$S_C = \{j_3, j_2, j_1, j_4\}$

$G(S_C) = 16$

$G(S_C) > G(S_3)$

$U_3 = 0.02$

$P(S_3, S_C) = exp\left\{\frac{G(S_3)-G(S_C)}{\beta_3}\right\} = 0.0643$

$U_3 < P(S_3, S_C)$

set $S_4 = S_C$, $S_4 = \{j_3, j_2, j_1, j_4\}$, $G(S_4) = 16$

$\beta_4 = (\beta_1)^4 = (0.9)^4 = 0.6561$

K = 4, N = 4

Let $S_C = \{j_1, j_3, j_2, j_4\}$

$G(S_C) = 15$

$G(S_0) < G(S_C) < G(S_4)$

set $S_5 = S_C$, $S_5 = \{j_1, j_3, j_2, j_4\}$, $G(S_5) = 15$

$\beta_5 = (\beta_1)^5 = 0.5905$

K = 5, N = 4,

$S_0 = \{j_2, j_1, j_3, j_4\}$, $G(S_0) = 14$,

5. Computational Experience of SA Algorithm

An intensive effort has been implemented in relation to numerical investigations. We will present in the next subsection how the problem under study is created randomly.

5.1 Test Problem [9]

There exists in the literature a classical way to randomly generate test problems of scheduling problems.

- The processing time a_j, b_j and c_j is uniformly distributed in the interval [1,10].
- The due date d_j is uniformly distributed in the interval [P(1-TF-RDD/2),P(1-TF+RDD/2)]; where $P = \sum_{j=1}^{n} p_j$ depending on the relative range of due date (RDD) and on the average tardiness factor (TF).
- The an integer weight of early job h_j were generated from uniform distribution [1,10].
- The an integer weighted of tardy job w_j were generated from uniform

distribution[1,10].

For both parameters, the values 0.2, 0.4, 0.6, 0.8 and 1.0 are considered. For each selected value of n (where n is the number of jobs), ten problems were generated.

5.2 Computational Results

In this section, we provide our results that were obtained for the simulated annealing algorithm with our proposed function. The test problems are provided in this paper. It's worth mentioning that we used the MATLAB 7.10.0 (R2010a) software for coding the algorithms and the processing personal computer was of 4 GB RAM with a CPU of 2.5 GHz speed.

Table (4) shows both the cost and times averages for 10 problems for each of its values from the number of jobs n where n ϵ {10, 50, 100, 500, 1000, 5000, 10000, 12000}.

Table (5) displays a calculated cost by **SA** algorithm for 10 problems at n= 500, the time of each problem and the number of iterations, in addition to the average of time and cost.

From the observation of the two tables (4) and (5) we find that the method of **SA**gives an approximate solution we can solve the problem with up to n = 12000 during 30 seconds, and **SA** algorithm will cease when a constant time amount is passed, here after 30 seconds.

Table abbreviations:

n	Number of jobs
No. of iterations	Number of iterations
No.	Number of problems
Av. of Cost	Average of cost
Av. of Time	Average of time for (10) examples in Simulated Annealing algorithm.

Table (4): The performance to average of cost and average of time in Simulated Annealing algorithm for n = {10,50,100,500,1000,5000,10000,12000}

n	Av. of Cost	Av. of Time
10	44.6000	0.2758
50	225.2000	2.8994
100	452.3000	9.0964
500	2292	29.1419
1000	4516.7	29.1454
5000	22482	29.4675
10000	45072	29.7156
12000	54291	30.4109

At this point, we find the solution to Simulated Annealing algorithm (SA) for n = 500.

Table (5): The Performance of Simulated Annealing (SA) Algorithm for n= 500

n	No.	Cost	Time	No. of Iterations
500	1	2307	29.0160	122
	2	2306	29.2483	160
	3	2270	29.1916	169
	4	2345	29.1700	170
	5	2186	29.0170	168
	6	2210	29.0610	163
	7	2408	29.2995	167
	8	2327	29.0784	168
	9	2271	29.1836	168
	10	2290	29.1533	166
Av. of Cost		**2292**		
Av. of Time		**29.1419**		

6. Conclusion

In this paper, we have developed approximate solutions for the problem of scheduling n independent jobs on one machine tominimizethesumof weighted number of earlyjobsand weighted number of tardy jobs under the fuzzy processing time.To resolve it we proposed the Average High Ranking method to obtain a processing time generated from fuzzy processing time, calculate the costs and reach to penalty cost.we report on the results of extensive computations tests of the method: simulated annealing. The main conclusion to be drawn from our computation results is that the SA algorithms can solved the problem(1/ /$\sum_{j=1}^{n} N_{h_j E_j} + \sum_{j=1}^{n} N_{w_j T_j}$)to 12000 jobs in 30 seconds.

References

[1] Gupta, S. and Rambha, M. (2011). Single Machine Scheduling With Distinct Due Dates Under Fuzzy Environment; International Journal of Enterprise Computing and Business Systems, vol. 1, no. 2, pp. 1–9.

[2] Baker, K.R. and Scudder, G.D. (1990). Sequencing with Earliness and Tardiness Penalties: A Review; Operations Research, Vol. 38, No.21, Pp. 22-36.

[3] Soroush, H. (2007). Minimizing the Weighted Number of Early and Tardy Jobs in a Stochastic Single Machine Scheduling Problem; European Journal of Operational Research, Vol. 181, No. 1, Pp. 266–287.

[4] Pinedo, M.L. (2016). Scheduling: Theory, Algorithms, and Systems; 5th Edition, Springer, New York.

[5] Al-Harkan, I.M. (1997). Algorithms for Sequencing and Scheduling; University of Riyadh, College of Engineering, Industrial Engineering Department, King Saud, Riyadh, Saudi Arabia, Ch. 8, Pp. 8-1, 8-19.

[6] Kirkpatrick, S. Gelatt Jr, C.D. and Vecchi, M.P. (1983). Optimization by Simulated Annealing; Science, Vol. 220, No. 4598, Pp. 671–680,

[7] Černý, V. (1985). Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm; Journal ofOptimization Theory and Applications, Vol. 45, No. 1, Pp. 41–51.

[8] Metropolis, N. Rosenbluth, A.W. Rosenbluth, M.N. Teller, A.H. and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines; The Journal of Chemical Physics, Vol. 21, No. 6, Pp. 1087-1092.

[9] Abdul-Razaq, T.S. Potts, C.N. and Van, Wassenhove. (1990). A survey of Algorithms for The Single Machine Total Weighted Tardiness Scheduling Problem; Discrete App. Math. Pp. 26235-253.