Minimization of earliness & Tardiness penalties with common due dates problem using Tabu Search

Conference Paper · March 2015

Minimization of earliness & Tardiness penalties with common due dates problem using Tabu Search

T.Aziz
Advanced Manufacturing Institute,
King Saud University, 11421, Riyadh, KSA
tariq_aziz_37@yahoo.com

Mohammed Hatem Mohammed Alhaag
Researcher King Saud University
University, 11421, Riyadh, KSA
inengmohamed@yahoo.com

Ibrahim M.Alharkan
Associate prof.IE Department
King Saud University, 11421, Riyadh, KSA
imalhark@ksu.edu.sa

Abstract— This research relates to a single machine scheduling problem with objective is to minimize earliness & Tardiness penalties with common due dates. We proposed a heuristic method for finding initial solution as an input to the Tabu search algorithm for finding a near optimal solution for this problem. In our research study the due dates are restrictive with due date parameter h, ranges from 0.2, 0.4, 0.6 to 0.8. Our study is a paper base in which we have applied the Tabu search algorithm for the prescribe problem. The base paper of (Ronconi and Kawamura 2010) studied single machine earliness and tardiness scheduling problem: lower bounds and a branch-and-bound algorithm) obtaining optimal solution. We compared the results of Tabu search algorithms with the base paper, Heuristic & benchmark results of D. Biskup and M. Feldmann. Our algorithm shows near optimal solution to the branch & bound algorithm.

Keywords—Tabu search; branch & bound; Restrictive due date; Single machine; Earliness; tardiness

I. INTRODUCTION

A. Scheduling and Sequencing

The increasing competition and ever-changing customer demands have been encouraging manufacturers to re-evaluate their manufacturing systems. The cost and on-time delivery have been the crucial factors for customers. Customers expect low cost products and on-time delivery from the manufacturers and for manufacturer, cost and customer satisfaction are the important parameters in order to stay in competition with other competitors. The key is to produce and deliver products to customers at the right time while minimizing the production costs as much as possible so that manufacturers can sell their products at a lower price than their competitors.

Scheduling plays a very important role in many manufacturing and production systems as well as information-processing environments. The need for scheduling also exists in transportation, distribution settings and in service industries. Scheduling can also be described as a decision-making process of allocating limited resources over time in order to perform a collection of tasks for the purpose of optimizing certain objective functions.
The instances of non-regular measures relate to due dates, such as the total of earliness and tardiness about a common due date. According to (Pinedo 2008) problem can be described by a triplet $\alpha | \beta | \gamma$. The α field describes the machine environment and contains only a single entry. The β field provides details of the processing characteristics and constraints. The γ field contains the objective to be minimized and usually contains only one single entry.

II. LITERATUREREVIEW

One of the pioneers studying common due date problems has been (Kanet 1981). He considered the problem of minimizing the sum of deviations from a common due date and presented a polynomially bounded matching algorithm which solves the problem in O *(nlogn)* time. This contribution has been extended in many directions; see, for example, (Biskup and Edwin Cheng 1999), (Hall and Posner 1991), (Hoogeveen and van de Velde 1991) and (Panwalkar, Smith, and Seidmann 1982). An excellent review is given by (Baker and Scudder 1990). When scheduling against a common due date some of the jobs may be completed early, that is, prior to the due date, while others are finished late.

According to JIT, earliness and tardiness are considered harmful to profitability and, for this reason, must be minimized: tardiness causes loss of customer goodwill and damage reputation, as well as delay of payments, while earliness causes inventory carrying costs and possible loss of product quality. Probably based on this motivation, many authors have considered the scheduling problem aiming to minimize earliness and tardiness in the delivery of goods. Comprehensive surveys on the common due date assignment and scheduling problems can be found in (Baker and Scudder 1990) and (Gordon, Proth, and Chu 2002). Therefore, criteria involving both earliness and tardiness costs are receiving increased attention in machine scheduling research (see the book by (Józefowska 2007)for a survey of models and algorithms in this area).

The goal is to find a schedule, which minimizes the sum of earliness and tardiness costs. This problem is NP-hard (Hall, Kubiak, and Sethi 1991); (Hoogeveen and van de Velde 1991). Although the exact classification is an open question, no pseudo polynomial algorithm is known and it is assumed that the problem is NP-hard in the strong sense (C.-Y. Lee, Danusaputro, and Lin 1991). Consequently, it has been tackled by meta-heuristic approaches, namely, tabu search (James 1997) and parallel genetic algorithms (C. Y. Lee and Kim 1995). As both approaches are marred by an underlying weakness, which will be discussed in connection with the properties of the problems, Lee & Kim develop a new and appropriate problem representation, which overcomes this weakness. In their study three metaheuristics, evolutionary search (ES), simulated annealing (SA) and threshold accepting (TA) using a new problem representation is presented. In addition, a new variant of TA, namely 'TA with a back step', is introduced. The approaches are implemented and tested extensively on benchmark problems recently given by (Biskup and Feldmann 2001) and (James 1997).

There are two main approaches to address the common due date. In the unrestricted case, the common due date is a decision variable or, if its value is known, it has no influence on the optimal sequence. This happens when the due date is greater than or equal to the sum of all processing times. However, if the due date is known and it affects the optimal sequence of jobs, then it is considered **restrictive.** (Kanet 1981) was one of the pioneers for the unrestrictive case, while (Cheng and Gupta 1989) made a survey about models in which due dates are decision variables.

III. PROBLEM DEFINITION AND CHARACTERISTICS

The problem of scheduling single machine in order to minimize total earliness & tardiness penalties against common due dates can be defined as

The problem can be defined as follows. There are n jobs available at time zero to be processed on a single machine and to be delivered on a common due date d. Each job i requires exactly one operation and its processing time pi is known. If a job i is completed before the due date, its earliness is given by $E_i = d - C_i$, where C_i is the completion time of job i. Conversely, if a job i is completed after the desired date, its tardiness is given by $T_i = C_i - d$. Each job i has its own unit earliness penalty α_i and unit tardiness penalty β_i. Preemption is not allowed and the initial processing time is not necessarily at time zero, when all jobs are available. The objective of the problem is to obtain an optimal schedule that minimizes the sum of earliness and tardiness penalties.

A. Common due date is unrestrictive:

If the optimal solution cost cannot decrease with the increase in the common due date (The common due date is greater than or equal to the sum of all the processing time).

B. Common due date is restrictive:

Due date affect the optimal solution; For the restrictive common due date case with general penalties, there is an optimal solution with these properties:

1. No idle times are inserted between consecutive jobs.
2. The schedule is V-shaped, that is, jobs that complete on or before the due date are sequenced in a non-increasing order of the p_i/α_i ratio. The jobs that start on or after the due date are sequenced in a non-decreasing order of the p_i/β_i ratio. Note that there may be a straddling job, i.e., a job whose processing is started before and finished after the due date.
3. There is an optimal schedule in which either the processing time of the first job starts at time zero or one job is completed on the due date.

The problem can be summarized as follows:

- Objective function: To minimize total Earliness &tardiness Penalties
- Scheduling environment: Single machines
- Constraints: restrictive common due date.

C. Problem formulation:

The following mixed integer linear programming (MILP) formulation can be used to obtain optimal solutions to this problem.

Parameters:

d**:** common due date;
α_i **:** earliness penalty of job i per time unit;
β_i **:** tardiness penalty of job i per time unit;
p_i **:** processing time of job i;
R**:** sufficiently large number.

Variables:

x_{ik} **:** 1, if job i is sequenced (not necessarily directly) prior to job k. 0, otherwise;

C_i **:** completion time of job i;
E_i **:** earliness of job i;
T_i **:** tardiness of job i.

Subject to

$T_i - E_i = C_i - d,\ i = 1, 2, \ldots, n,$ (2)

$C_i \leq C_k - p_k + R\,(1 - x_{ik}), i = 1, 2, \ldots, n-1, k = i+1, \ldots, n,$ (3)

$C_k \leq C_i - p_i + R\, x_{ik},\ i = 1, 2, \ldots, n-1, k = i+1, \ldots, n,$ (4)

$C_i - p_i \geq 0,\ i = 1, 2, \ldots, n,$ (5)

$T_i \geq 0,\ i = 1, 2, \ldots, n,$ (6)

$E_i \geq 0,\ i = 1, 2, \ldots, n,$ (7)

$x_{ik}\, 2\ \{0, 1\}, i = 1, 2, \ldots, n-1, k = i+1, \ldots, n.$ (8)

Equation (1) represents the objective function to be minimized, i.e., the sum of tardiness and earliness penalties. In [3] the tardiness and earliness are calculated through the following restrictions: $T_i \geq d - C_i,\ i = 1, 2, \ldots, n,$
$E_i \geq C_i - d,\ i = 1, 2, \ldots, n.$

Alternatively, we calculate tardiness and earliness through the constraints of form (2). Note that the presented model has n restrictions less than the original one. Constraints of form (3) and (4) indicate the completion time of each job: if job i is sequenced prior to job k, $x_{ik} = 1$ and, consequently, restriction (3) gives $C_i \leq C_k - p_k$ and, due the addition of constant R, restriction (4) is not restrictive. On the other hand, if $x_{ik} = 0$, restriction (4) becomes $C_k \leq C_i - p_i$ and restriction (3) is not restrictive. Restriction (5) assures that initial time of each job i is not negative. The set of restrictions (6) and (7) defines the non-negativity of variables T_i and E_i, while restriction (8) defines the variable x_{ik} as binary.

IV. PURPOSED HEURISTIC

Since at most one job can be completed on the due date, there will be a set of jobs (S^E) finished before the due date and a set of jobs (S^T) finished after the due date. The heuristic strategy, here in after called HRM, consists of: (i) determining these two sets, (ii) constructing a sequence for each set, and (iii) setting the final scheduling S as the concatenation of both sequences. To make sure that S will satisfy properties (1) and (2), there will be no idle time between Consecutive jobs, and the sequences of (S^E) and (S^T)will be V-shaped'' and L-shaped'', respectively. At each iteration, the non-scheduled jobs with the maximum ratios pj/aj and pj/bj are considered for inclusion in one of the two sets. According to the distance between each job's possible completion time and the due date, just one of the jobs is included.

Notations:

P: Set of jobs to be allocated
g: Idle time inserted at the beginning of the schedule
S^E Set of jobs completed on the due date or earlier
S^T Set of jobs completed after the due date
S Schedule representation $S = (g, S^E; S^T)$
e Candidate job for S^E
t Candidate job for S^T
E^e Distance between the possible completion time of job e and the due date
T^t Distance between the possible completion time of job t and the due date
d^T Time window available for inserting a job in set ST
d^E Time window available for inserting a job in set SE
H Total processing time, $H = \sum_{j=1}^{n} Pj$

A. Algorithm based on proposed heuristic for initial solution

Procedure or Propositions:

Step 1: Let P= (1; 2; . . . ; n), $S^E = S^T = \{\Phi\}$, $d^E = d$, $d^T = H - d$.

Step 2: Set e = arg max $j \in p \left\{\frac{Pj}{\alpha j}\right\}$ and t= e=arg max $j \in p \left\{\frac{Pj}{\beta j}\right\}$ (in case of tie, select the job with the longest pj).

Step 3: Set $E^e = d^E - Pe$ and $T^t = d^T$.

Step 4: Choose the job to be inserted:

If $E^e > T^t$ then $S^E = S^E + \{e\}$ and $d^E = d^E - pe$ and $P = P - \{e\}$.

If $E^e < Tt$ then $S^T = S^T + \{t\}$ and $d^T = d^T - Pt$ and $P = P - \{t\}$.

If $E^e = T^t$ then if $\alpha e > \beta t$ then $S^T = S^T + \{t\}$ and $d^T = d^T - Pt$ and $P = P - \{t\}$ else $S^E = S^E + \{e\}$, $d^E = d^E - Pe$ and $P = P - \{e\}$

Step 5: If $P \neq \{\varphi\}$ then go to step 2.

Step 6: If there is a job k (straddling job) that has part of its processing time (Pb > 0) before the due date and the remaining part after the due date (Pa > 0) then apply g-test:

$$g' = g + Pb \qquad S' = (g', SE, ST)$$

$$g'' = Max\{0, g - Pa\}, \qquad S'' = (g'', SE + \{k\}, ST - \{k\});$$

$S = argmin\{f(s), f(S'), f(S'')\}$. *End Of algorithm*

It should be mentioned that an insertion in S^E is always made at the end of the set. For instance, if job {2} is inserted in set S^E\{1; 4; 3\}, it yields a new set S^E\{1; 4; 3; 2\}. On the other hand, when the insertion is made in set S^T, the new job is placed at the beginning of the set. Furthermore, every calculation of $f(S)$ follows property (2). To make this evaluation it is important to note that set S^T can have a straddling job. If no straddling job is present, all elements of set S_E are ordered in a non-increasing order of ratio pj/αj and the elements of S^T are sequenced in a non-decreasing order of ratio pj/βj. If there is a straddling job in the set, the following procedure is used:

Step 1: Order set S^E in a non-increasing order of ratio pj/αj.
Step 2: Order set S^T in a non-decreasing order of ratio pj/βj. Note that now the straddling job may be another job or it may even disappear.
Step 3: If there are jobs that belong to set S^T and they are now completed before or on the due date, put these jobs on set S^E following the ''\-shape'' format. Stop.

The algorithm starts with two copies of P, ordered by ratios pj/αj and pj/βj, and maintains the sequence S^E (S^T) in non-increasing (non-decreasing) order of pj/αj (pj/βj). Then, it is easy to verify that each iteration can be implemented with O(n) operations and that the complete algorithm has complexity O(n2). Note that, at the end of the algorithm, a unique solution will be represented by the variable g and sequences S^E and S^T.

B. Tabu Search

1. General Introduction

Building upon some of his previous work, (F. W. Glover and Laguna 1997) proposed a new approach, which he called Tabu Search, to allow LS (Local Search) methods to overcome local optima. (In fact, many elements of this first TS proposal, and some elements of later TS elaborations, were introduced in (F. Glover and Laguna 1977), including short term memory to prevent the reversal of recent moves, and longer term frequency memory to reinforce attractive components.) The basic principle of TS is to pursue LS whenever it encounters a local optimum by allowing non-improving moves; *cycling* back to previously visited solutions is prevented by the use of *memories*, called *Tabu lists*, that record the recent history of the search, a key idea that can be linked to Artificial Intelligence concepts. It is interesting to note that, the same year, Hansen proposed a similar approach, which he named *steepest ascent/mildest descent*. It is also important to remark that Glover did not see TS as a proper heuristic, but rather as a Meta-Heuristic, i.e., a general strategy for guiding and controlling "inner" heuristics specifically tailored to the problems at hand. Tabu Search (TS) is among the most used Meta heuristic. TS explicitly uses the history of the search, both to escape from local minima and to implement an explorative strategy. A simple version of TS will be described, to introduce the basic concepts. In tabu search we move from the one schedule to another with the next schedule being possibly worse than the one before. For each schedule a neighborhood is defined. The search within the neighborhood for a potential candidate to move to be again a design issue. In tabu-search the mechanism is not probabilistic but rather of a deterministic nature. At any stage of the process a tabu-list of mutations, which the procedure is *not* allowed to make, is kept.

Step 1:
Get an initial sequence from the purpose heuristic S_1
Initialize: Set K=1
Suppose So=S1
Therefore Z (So) =Z (S1)
Step 2:
Select S_c from the neighbourhood of S_k

If the move from the S_c to S_k is not allowed in tabu List,
Then Go To Step 3
IF $Z(S_c) \leq Z(S_o)$ Then S_o=S_c Then Delete the previous tabu List & Add fresh Move in tabu List GO TO Step (3)
Step 3:
Set k=k+1
If the $k \leq N$ Then repeat the Step 2 Otherwise Stop

C. Comparison between the results of Tabu Search & Initial Solution from the purposed Heuristic

We have three different number of jobs {n=10, 20, 50} and four due date restrictive factors h=0.2, 0.4, 0.6 and 0.8 are considerate in this study. The factor h indicates that how jammed the production line is at the beginning of the schedule & It is used in the definition of the common due date. According to the expression $d = h \sum_{i=1}^{n} P_i$. The processing time, earliness penalty & tardiness penalty are taken from the benchmark of D. Biskup and M. Feldmann for "scheduling on a single-machine against restrictive and unrestrictive common due dates". The link for the benchmark data regarding processing time, earliness penalty & tardiness penalty is given below.

http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/sch10.txt
[n=10]

http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/sch20.txt
[n=20]

http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/sch50.txt
[n=50]

TABLE -1: COMPARISON BETWEEN PURPOSED HEURISTIC & TS AT N=10

n=10	h=.2			h=.4			h=.6			h=.8		
	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT
K=1	1936	1936	0.129	1457	1025	0.109	860	860	0.109	1026	1022	0.125
K=2	1748	1001	0.109	1548	628	0.109	1111	877	0.109	1432	1432	0.109
K=3	1731	1586	0.109	1510	917	0.094	1010	933	0.109	1383	1301	0.094
K=4	2183	2139	0.109	1229	1180	0.125	815	815	0.062	952	952	0.078
K=5	1443	1149	0.125	905	619	0.094	627	550	0.109	873	820	0.218
K=6	1544	1469	0.109	1739	908	0.125	940	770	0.14	997	904	0.109
K=7	2550	2102	0.109	2096	1374	0.109	1581	1083	0.109	1501	1283	0.109
K=8	1971	1680	0.094	1103	1003	0.109	614	610	0.109	665	560	0.109
K=9	1913	1574	0.109	1114	876	0.125	734	583	0.125	631	596	0.125
K=10	1913	1869	0.109	1119	1097	0.109	710	710	0.094	899	691	0.125
Mean	1893.2	1650.5	0.111	1382	962.7	0.111	900.2	779.1	0.108	1035.9	956.1	0.12
S.D	299.34121	359.0146	0.009	340.08558	219.69982	0.011	274.00504	160.578	0.019	291.39851	288.91918	0.036
Improvement from Intial solution		14.7046%			43.5546%			15.54358%			8.34641%	

Table -1 shows that there are 10 instances are to be tested for each problem size & for each restrictive factor, totaling 3x4x10 are equal to 120 problems. The computer code written in a Matlab language. The experiments run on the "Processor Intel (R) Core I3 (TM) CPU, 3GB, 32 bits OS configuration". The table below represents the results Initial solution obtained from Purposed Heuristic method & Tabu search. The CPU time obtains in each instance corresponding to each restrictive factor. The percentage difference is computed for each restrictive factor for Purposed heuristic & TS. At h=.2 the tabu search gives 14.7% & at h= 0.4 the improvement of TS is 43.556%, at h=0.6 the TS improvement 15.5435% and at h=0.8 TS gives 8.3464 % more accurate results as compared to the purposed heuristic.

TABLE: 2 COMPARISON BETWEEN PURPOSED HEURISTIC & TS AT N=20

n=20	h=.2			h=.4			h=.6			h=.8		
	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT
K=1	6497	4394	0.562	4273	3077	0.577	3992	3246	0.577	5117	4809	0.562
K=2	8803	8430	0.577	5644	4803	0.562	3399	3214	0.562	3485	3417	0.577
K=3	6214	6143	0.577	5539	3833	0.593	5183	3885	0.562	5759	5540	0.593
K=4	9632	9188	0.562	6031	5118	0.53	3548	3317	0.562	3579	3419	0.577
K=5	5053	4215	0.577	3165	2495	0.772	3000	2204	0.562	3176	3055	0.624
K=6	6660	6527	0.562	5443	3539	0.577	4797	3111	0.562	5614	4861	0.562
K=7	11224	10342	0.577	7112	6195	0.562	4431	4126	0.608	4380	3878	0.562
K=8	4850	3920	0.562	3594	2096	0.562	2512	1721	0.562	2272	2117	0.562
K=9	3600	3404	0.562	3136	2080	0.562	2355	2067	0.562	3055	2801	0.562
K=10	6259	4979	0.593	4220	2973	0.562	3469	2095	0.54	3280	2640	0.562
Mean	6879.2	6154.2	0.571	4815.7	3620.9	0.586	3668.6	2898.6	0.566	3971.7	3653.7	0.574
S.D	2222.234857	2299.024	0.01	1266.5241	1303.8136	0.064	888.251	782.712	0.016	1122.5867	1048.161	0.019
Improvement from Intial solution		11.781%			32.997%			26.565%			8.704%	

Table -2 shows that there are 20 instances are to be tested for each problem size & for each restrictive factor, at h=.2 the tabu search gives 11.781% & at h= 0.4 the improvement of TS is 32.997%, at h=0.6 the TS improvement 26.565% and at h=0.8 TS gives 8.704 % more accurate results as compared to the purposed heuristic.

TABLE-3: COMPARISON BETWEEN PURPOSED HEURISTIC & TS AT N=50

n=50	h=.2			h=.4			h=.6			h=.8		
	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT	Initial Solution	TS solution	CPUT
K=1	45326	40570	2.746	27799	24028	2.792	22753	18013	2.761	23367	22165	2.792
K=2	38264	30629	2.87	26268	17837	2.824	20307	14133	2.808	19871	17765	2.777
K=3	42570	34470	2.184	25617	20732	2.714	20581	16679	2.761	24127	23300	2.184
K=4	33409	27616	2.746	26173	16675	2.761	21915	14722	2.761	22772	20183	2.792
K=5	35050	32371	2.746	21744	17957	2.855	16722	15134	2.73	22108	21419	2.902
K=6	43881	34835	2.73	26143	20541	2.792	19099	14501	2.777	20768	17890	2.746
K=7	45055	42900	2.714	28502	23064	2.699	21539	17713	2.699	25791	23928	2.777
K=8	54297	43665	2.699	40203	24918	2.792	29163	21999	2.761	32678	30277	2.777
K=9	41434	34268	2.683	24195	20002	2.683	16337	14322	2.761	17112	16537	2.668
K=10	36090	33008	2.699	24983	19211	2.808	18352	14395	2.824	21816	18968	2.777
Mean	41537.6	35433.2	2.682	27162.7	20496.5	2.772	20676.8	16161.1	2.764	23041	21243.2	2.719
S.D	5870.693029	5019.893	0.173	4694.4397	2620.9455	0.054	3480.1884	2380.83	0.033	3931.0544	3817.3216	0.186
Improvement from Intial solution		17.228%			32.524%			27.942%			8.463%	

Table -3 shows the comparison between the Purposed heuristic & TS at n=50 having number of jobs 50, h=.2 the tabu search gives 17.228% & at h= 0.4 the improvement of TS is 32.524%, at h=0.6 the TS improvement 27.942% and at h=0.8 TS gives 8.463 % more accurate results as compared to the purposed heuristic.

TABLE-4: COMPARISON BETWEEN AUTHOR & BENCHMARK RESULT OF D. BISKUP AND M. FELDMANN AT N=20

Comparision of results of Author & Bench mark(Author Lower bound,Bench mark upper bound) N=20												
	h = 0.2			h=.4			h=0.6			h=0.8		
K	Optimal valu Author LB	Optimal valu Bench mark	% Differenc	Optimal valu Author LB	Optimal valu Bench mark	% Difference	Optimal valu Author LB	Optimal valu Bench mark	% Difference	Optimal valu Author LB	Optimal valu Bench mark	% Difference
1	4394	4431	0.8421%	3066	3066	0.000%	2986	2986	0.0000%	2986	2986	0.000%
2	8430	8567	1.6251%	4847	4897	1.032%	3206	3260	1.6843%	2980	2980	0.000%
3	6210	6331	1.9485%	3838	3883	1.172%	3583	3600	0.4745%	3583	3600	0.474%
4	9188	9478	3.1563%	5118	5122	0.078%	3317	3336	0.5728%	3040	3040	0.000%
5	4215	4340	2.9656%	2495	2571	3.046%	2173	2206	1.5186%	2173	2206	1.519%
6	6527	6766	3.6617%	3582	3601	0.530%	3010	3016	0.1993%	3010	3016	0.199%
7	10455	11110	6.2649%	6238	6357	1.908%	4126	4175	1.1876%	3878	3900	0.567%
8	3920	4203	7.2194%	2145	2151	0.280%	1638	1638	0.0000%	1638	1638	0.000%
9	3465	3530	1.8759%	2096	2097	0.048%	1965	1992	1.3740%	1965	1992	1.374%
10	4979	5545	11.3677%	2925	3192	9.128%	2110	2116	0.2844%	1995	1995	0.000%
Average	6178.3	6430.1	4.0756%	3635	3693.7	1.615%	2811.4	2832.5	0.7505%	2724.8	2735.3	0.385%

Table -4 shows the comparison between the authors with the bench marked results having number of jobs 20, with 40 instances generated by branch & bound algorithm. He also computed the percentage difference of the optimal value F_0 in relation to the benchmark value of **D. Biskup and M. Feldmann as follows % difference ((F_{BF}-F_0)/F_0)*100** where F_{BF} is D. Biskup and M. Feldmann bench mark value & F_0 is optimal value obtained by Author from branch & bound. The above table shows the percentage difference for each consider problem. It is observed that in a worst case scenario the benchmark value is 11.37% greater than the optimal value obtain from branch & bound. The optimal value obtain from author reveals that these preference values can be improved by 11.37% .These results can be used to evaluate the performance of Heuristic& meta Heuristic for the consider problem.

TABLE-5: COMPARISON OF RESULT OF AUTHOR & TABU SEARCH (AUTHOR LOWER BOUND, TABU SEARCH) N=20

Comparision of results of Author & Tabu Search (Author Lower bound,Tabu Search) N=20												
	h = 0.2			h=.4			h=0.6			h=0.8		
K	Optimal value Author LB	Optimal value tabu search	% Differenc	Optimal value Author LB	Optimal value tabu search	% Difference	Optimal value Author LB	Optimal value tabu search	% Difference	Optimal value Author LB	Optimal value tabu search	% Difference
1	4394	4394	0.0000%	3066	3077	0.359%	2986	3246	8.7073%	2986	4809	61.052%
2	8430	8430	0.0000%	4847	4847	0.000%	3206	3214	0.2495%	2980	3417	14.664%
3	6210	6243	0.5314%	3838	3843	0.130%	3583	3885	8.4287%	3583	5540	54.619%
4	9188	9188	0.0000%	5118	5118	0.000%	3317	3317	0.0000%	3040	3419	12.467%
5	4215	4215	0.0000%	2495	2495	0.000%	2173	2204	1.4266%	2173	3055	40.589%
6	6527	6527	0.0000%	3582	3599	0.475%	3010	3111	3.3555%	3010	4861	61.495%
7	10455	10455	0.0000%	6238	6195	-0.689%	4126	4126	0.0000%	3878	3878	0.000%
8	3920	3920	0.0000%	2145	2196	2.378%	1638	1721	5.0672%	1638	2117	29.243%
9	3465	3504	1.1255%	2096	2096	0.000%	1965	2067	5.1908%	1965	2801	42.545%
10	4979	4979	0.0000%	2925	2973	1.641%	2110	2095	-0.7109%	1995	2640	32.331%
Average	6178.3	6185.5	0.1165%	3635	3643.9	0.245%	2811.4	2898.6	3.1017%	2724.8	3653.7	34.091%

Our contribution to the study shows that Tabu search gives near optimal solution. From the above table-4 it is clear that Branch & bound has better improvement with respect to tabu search. Tabu search gives near optimal solution when h=0.2 & 0.4 respectively. But at h=0. 6 & 0.8 the branch & bound is overall optimal and Tabu search is forward away from the optimal solution. At h=0.2 B&B is 0.1165% optimal as compared to tabu search and at h=0.4, B&B is optimal about 0.245%. But at h=0.6 & 0.8 B&B is better than Tabu search. In the worst case the tabu search is away about 34.091% from the optimal solution at h=0.8 respectively.

TABLE-6: COMPARISON OF RESULT OF TABU SEARCH & BENCHMARK OF D.BISKUP AND M.FELDMANN (TABU SEARCH, UPPER BOUND) N=10

Comparison of results of Tabu Search & Benchmark of D. Biskup and M. Feldmann (Upper bound, Tabu Search) N=10												
	h = 0.2			h=.4			h=0.6			h=0.8		
K	Optimal value TS	Benchmark value	% Difference	Optimal value TS	Benchmark value	% Difference	Optimal value TS	Benchmark value	% Difference	Optimal value TS	Benchmark value	% Difference
1	1936	1936	0	1025	1025	0	860	841	-2.2093023	1022	818	-19.960861
2	1001	1042	4.0959041	628	615	-2.07006	877	615	-29.874572	1432	615	-57.053073
3	1586	1586	0	917	917	0	933	793	-15.005359	1301	793	-39.046887
4	2139	2139	0	1180	1230	4.237288	815	815	0	952	803	-15.651261
5	1149	1187	3.3072237	619	630	1.77706	550	521	-5.2727273	820	521	-36.463415
6	1469	1521	3.539823	908	908	0	770	755	-1.9480519	904	755	-16.482301
7	2102	2170	3.2350143	1374	1374	0	1083	1101	1.6620499	1283	1101	-14.185503
8	1680	1720	2.3809524	1003	1020	1.694915	610	610	0	560	540	-3.5714286
9	1574	1574	0	876	876	0	583	582	-0.1715266	596	554	-7.0469799
10	1869	1869	0	1097	1136	3.55515	710	710	0	691	671	-2.894356
average	1651	1674.4	1.44805%	962.7	973.1	1.0803%	779.1	734.3	-5.7502%	956.1	717.1	-24.99739%
S.D	378.4	372.49465		219.7	228.2356		160.578	159.75171		288.91918	167.66243	

The above table-6 shows that Tabu search gives near optimal solution. From the above table it is clear that Tabu Search has better when h=0.2(1.448%)&0.4(1.0803%), But at h=0.6 &0.8 the Benchmark results of D. Biskup and M. Feldman is overall optimal and Tabu search is for away from optimal solution about to 5.750% & 24.99% respectively.

TABLE-7: COMPARISON OF RESULT OF TABU SEARCH & BENCHMARK OF D.BISKUP AND M.FELDMANN (TABU SEARCH, UPPER BOUND) N=20

Comparison of results of Tabu Search & Benchmark of D. Biskup and M. Feldmann (Upper bound, Tabu Search) N=20												
	h = 0.2			h=.4			h=0.6			h=0.8		
K	Optimal valu TS	Benchmark value	% Difference	Optimal valu TS	Benchmark value	% Difference	Optimal valu TS	Benchmark value	% Difference	Optimal valu TS	Benchmark value	% Difference
1	4394	4431	0.8420574	3077	3066	-0.35749	3246	2986	-8.0098583	4809	2986	-37.908089
2	8430	8567	1.6251483	4847	4897	1.031566	3214	3260	1.4312383	3417	2980	-12.788996
3	6243	6331	1.4095787	3843	3883	1.040853	3885	3600	-7.3359073	5540	3600	-35.018051
4	9188	9478	3.1562908	5118	5122	0.078156	3317	3336	0.5728068	3419	3040	-11.085113
5	4215	4340	2.9655991	2495	2571	3.046092	2204	2206	0.0907441	3055	2206	-27.790507
6	6527	6766	3.6617129	3599	3601	0.055571	3111	3016	-3.0536805	4861	3016	-37.955153
7	10455	11110	6.264945	6195	6357	2.615012	4126	4175	1.1875909	3878	3900	0.56730273
8	3920	4203	7.2193878	2196	2151	-2.04918	1721	1638	-4.8227775	2117	1638	-22.626358
9	3504	3530	0.7420091	2096	2097	0.04771	2067	1992	-3.628447	2801	1992	-28.882542
10	4979	5545	11.367745	2973	3192	7.366297	2095	2116	1.0023866	2640	1995	-24.431818
Average	6186	6430.1	3.9544095	3643.9	3693.7	1.366668	2898.6	2832.5	-2.2804112	3653.7	2735.3	-25.136163
S.D	2308	2416.589		1294.37	1321.4048		782.712	769.67594		1048.161	705.92918	

The above table-7 shows that Tabu search gives near optimal solution. From the above table it is clear that Tabu Search has better when h=0.2 (3.95%) & 0.4 (1.366%), But at h=0.6 &0.8 the Benchmark results of D. Biskup and M. Feldman is overall optimal and

Tabu search is for away from optimal solution about to 2.280% & 25.136% respectively.

TABLE-8: COMPARISON OF RESULT OF TABU SEARCH & BENCHMARK OF D.BISKUP AND M.FELDMANN (TABU SEARCH, UPPER BOUND) N=50

Comparison of results of Tabu Search & Benchmark of D. Biskup and M. Feldmann (Upper bound, Tabu Search) N=50												
K	h = 0.2			h=.4			h=0.6			h=0.8		
	Optimal valu TS	Benchmark value	% Difference	Optimal valu TS	Benchmark value	% Difference	Optimal valu TS	Benchmark value	% Difference	Optimal valu TS	Benchmark value	% Difference
1	40570	42363	4.4195218	24028	24868	3.495921	18013	17990	-0.1276856	22165	17990	-18.836003
2	30629	33637	9.8207581	17837	19279	8.084319	14133	14231	0.6934126	17765	14132	-20.450324
3	34470	37641	9.1993037	20732	21353	2.995369	16679	16497	-1.0911925	23300	16497	-29.197425
4	27616	30166	9.2337775	16675	17495	4.917541	14722	14105	-4.1910067	20183	14105	-30.114453
5	32371	32604	0.7197801	17957	18441	2.695328	15134	14650	-3.198097	21419	14650	-31.602783
6	34835	36920	5.9853596	20541	21497	4.654106	14501	14251	-1.724019	17890	14075	-21.324762
7	42900	44277	3.2097902	23064	23883	3.550989	17713	17715	0.0112911	23928	17715	-25.965396
8	43665	46065	5.496393	24918	25402	1.942371	21999	21367	-2.8728579	30277	21632	-28.553027
9	34268	36397	6.2127933	20002	21929	9.634037	14322	14298	-0.1675744	16537	13979	-15.468344
10	33008	35797	8.4494668	19211	20048	4.356879	14395	14298	-0.6738451	18968	14377	-24.203922
average	35433	37586.7	6.078%	20496.5	21419.5	4.5032%	16161.1	15940.2	-1.3669%	21243.2	15915.2	-25.08%
S.D	5020	4899.1305		20496.5	21419.5		2380.83	2308.9556		3817.3216	2410.871	

The above table-8 shows that Tabu search gives near optimal solution. From the above table it is clear that Tabu Search has better when h=0.2 (6.078%) & h= 0.4 (4.5032%), But at h=0.6 & 0.8 the Benchmark results of D. Biskup and M. Feldman is overall optimal and Tabu search is for away from optimal solution about to 1.3669% & 25.08% respectively.

V. CONCLUSION:

This paper considered the single machine scheduling problem with restrictive common due date involving tardiness and earliness penalties. This type of problem became more important with the advent of the lean production principle, including the just in time (JIT) concept. Due to its complexity, most of the authors addressed this problem using heuristic and Meta heuristic approaches.

In this study, a tabu search algorithm was proposed to find the near optimal solution to this problem. In the development of the algorithm, the use of the problem properties was important for the development of the tube search

Results are analyzed for 120 instances of prescribing problem and tested with the bench mark & base paper. The purposed Algorithm Tabu search gives near optimal solution when restrictive due date parameter h=0.2 & 0.4 is considered. When the restrictive parameter for due date is increases the Tabu search does not remain optimal. Branch & bound gives optimal results at all studied restrictive due date parameters (h=0.2, 0.4, 0.6 &0.8).

In this study a comparison is made between four different scenarios at n=10, 20, & 50) and due date restrictive parameter (h=0. 2, 0.4, 0.6 &0. 8).

1. Comparison between the results of Tabu Search & Initial Solution from the purposed Heuristic.
2. Comparison of results of Author & Benchmark (Author Lower bound, Bench mark upper bound).
3. Comparison of results of Author & Tabu Search (Author Lower bound, Tabu Search)
4. Comparison of results of Tabu Search & Benchmark of D. Biskup and M. Feldmann (Upper bound, Tabu Search).

In the first case Tabu search gives optimal as compared to the purposed heuristic &case author has more optimal results as compared to benchmark results, Case 3& 4 Tabu search is near optimal at h=0. 2 & 0.4 and does not remain too near Optimal at 0.6 & 0.8.

VI. FUTURE RECOMMENDATIONS:

The future study of this problem can be done by using different integrating approaches & with best initial solution heuristic would produce better outcomes for comparison with the benchmark results.

ACKNOWLEDGEMENT

Advanced manufacturing institute at King Saud University and KACST (King Abdulaziz City for Science and Technology) is highly acknowledge for continuous financial support of the present work.

REFERENCES:

1. Baker, Kenneth R., and Gary D. Scudder. 1990. "Sequencing with Earliness and Tardiness Penalties: A Review".Research-article.February1. http://pubsonline.informs.org/doi/abs/10.1287/opre.38.1.22.
2. Biskup, Dirk, and T.C. Edwin Cheng. 1999. "Multiple-Machine Scheduling with Earliness, Tardiness and Completion Time Penalties." *Computers & Operations Research* 26 (1): 45–57. doi:10.1016/S0305-0548(98)00044-6 http://linkinghub.elsevier.com/retrieve/pii/S0305054898000446.
3. Biskup, Dirk, and Martin Feldmann. 2001. "Benchmarks for Scheduling on a Single Machine against Restrictive and Unrestrictive Common Due Dates." *Computers & Operations Research* 28(8):787–801.doi:10.1016/S0305-0548(00)00008-3. http://www.sciencedirect.com/science/article/pii/S0305054800000083.
4. Cheng, T. C. E., and M. C. Gupta. 1989. "Survey of Scheduling Research Involving Due Date Determination Decisions." *European Journal of Operational Research* 38 (2): 156–66. http://ideas.repec.org/a/eee/ejores/v38y1989i2p156-166.html.
5. Glover, Fred, and Manuel Laguna. 1997. *Tabu Search*. Springer.
6. Glover, Fred W., and Manuel Laguna. 1997. *Tabu Search*. Springer.
7. Gordon, Valery, Jean-Marie Proth, and Chengbin Chu. 2002. "A Survey of the State-of-the-Art of Common due Date

Assignment and Scheduling Research.” *European Journal of Operational Research* 139 (1): 1–25. Doi:10.1016/S0377-2217(01)00181-3. http://www.emeraldinsight.com/bibliographic_databases.htm?id=1356720&PHPSESSID=kjb1qohcckfnurujfs42bplmj2.
8. Hall, Nicholas G., Wieslaw Kubiak, and Suresh P. Sethi. 1991. “Earliness–Tardiness Scheduling Problems, II: Deviation of Completion Times about a Restrictive Common Due Date”. Research-article. October 1. http://pubsonline.informs.org/doi/abs/10.1287/opre.39.5.847.
9. Hall, Nicholas G., and Marc E. Posner. 1991. “Earliness-Tardiness Scheduling Problems, I: Weighted Deviation of Completion Times about a Common Due Date”. Research-article. October 1. http://pubsonline.informs.org/doi/abs/10.1287/opre.39.5.836.
10. Hoogeveen, J. A., and S. L. van de Velde. 1991. “Scheduling around a Small Common Due Date.” *European Journal of Operational Research* 55 (2): 237–42. Doi:10.1016/0377-2217(91)90228-N. http://www.sciencedirect.com/science/article/pii/037722179190228N.
11. James, R. J. W. 1997. “Using Tabu Search to Solve the Common Due Date Early/Tardy Machine Scheduling Problem.” *Comput. Oper. Res.* 24 (3): 199–208. Doi:10.1016/S0305-0548(96)00052-4. http://dx.doi.org/10.1016/S0305-0548 (96)00052-4.
12. Józefowska, Joanna, ed. 2007. “Just-in-Time Concept in Manufacturing and Computer Systems.” In *Just-In-Time Scheduling: Models and Algorithms for Computer and Manufacturing Systems*, 1–23. International Series In Operations Research Amp; Mana 106. Springer US. http://link.springer.com/chapter/10.1007/978-0-387-71718-0_1.
13. Kanet, John J. 1981. “Minimizing the Average Deviation of Job Completion Times about a Common Due Date.” *Naval Research Logistics Quarterly* 28 (4): 643–51. doi:10.1002/nav.3800280411. http://onlinelibrary.wiley.com/doi/10.1002/nav.3800280411/abstract.
14. Lee, Chae Y., and Seok J. Kim. 1995. “Parallel Genetic Algorithms for the Earliness-Tardiness Job Scheduling Problem with General Penalty Weights.” *Comput. Ind. Eng.* 28 (2): 231–43. doi:10.1016/0360-8352(94)00197-U. http://dx.doi.org/10.1016/0360-8352(94)00197-U.
15. Lee, Chung-Yee, Surya Liman Danusaputro, and Chen-Sin Lin. 1991. “Minimizing Weighted Number of Tardy Jobs and Weighted Earliness-Tardiness Penalties about a Common Due Date.” *Computers & Operations Research* 18 (4): 379–89. doi:10.1016/0305-0548(91)90098-C. http://www.sciencedirect.com/science/article/pii/030505489190098C.
16. Panwalkar, S. S., M. L. Smith, and A. Seidmann. 1982. “Common due Date Assignment to Minimize Total Penalty for the One Machine Scheduling Problem”. Research-article. April 1. http://pubsonline.informs.org/doi/abs/10.1287/opre.30.2.391.
17. Pinedo, Michael. 2008. *Scheduling: Theory, Algorithms, and Systems*. Springer.
18. Ronconi, Débora P., and Márcio S. Kawamura. 2010. “The Single Machine Earliness and Tardiness Scheduling Problem: Lower Bounds and a Branch-and-Bound Algorithm.” *Computational & Applied Mathematics* 29: 107–24.

Tariq Aziz is a Researcher in Advance Manufacturing Institute, King Saud University (Riyadh, Saudi Arabia). He received his BSc in Industrial Engineering and Management from the University of Punjab (Lahore, Pakistan) and his MSc in Manufacturing System Engineering from the King Saud University (Riyadh, Saudi Arabia). He has diverse expertise in human-computer interaction, quality engineering and simulating human machine systems. He previously worked on projects related to the application of Six Sigma and risk management in entertainment industry of Saudi Arabia. He is recently working on ‘biomedical feet joint implant for rheumatoid patients’ and ‘multi task performance assessment’ (CAN, MAP).

Ibrahim M Alharkan is an Associate Professor and Chairman of deanship of admission at King Saud University, Research interests are in the areas of production planning and control, modeling and simulation of industrial and service systems, and applied operations research. These areas of interest include production planning and control, inventory control, production sequencing, scheduling and lot sizing; expert system; simulated annealing algorithms; genetic algorithms; Tabu search; scatter search algorithms, and total quality management, quality control, maintenance planning and scheduling, project scheduling.

Mohammed Hatem Alhaag is a Researcher at King Saud University, His area of expertise is human factor engineering. He received his BSc in Industrial Engineering from the University of Taiz (Taiz, Yemen) and his MSc in Human Factor Engineering from the King Saud University (Riyadh, Saudi Arabia). He has diverse expertise in human-computer interaction, quality engineering and simulating human machine systems.

.

APPENDIX:

The proposed algorithm (Tabu Search) to solve given problem

```
%Main Function for Tabu search%
clear
close all
clc
NOJ=input('Please Enter the Number of Jobs = ');
```

```
H=input('Please Enter the control jamping = ');
switch NOJ
case 10
       AA=importdata('n10.txt');
       p=AA.data(:,1);
       a=AA.data(:,2);
       b=AA.data(:,3);
case 20
       AA=importdata('n20.txt');
       p=AA.data(:,1);
       a=AA.data(:,2);
       b=AA.data(:,3);
case 50
       AA=importdata('n50.txt');
       p=AA.data(:,1);
       a=AA.data(:,2);
       b=AA.data(:,3);
case 100
       AA=importdata('n100.txt');
       p=AA.data(:,1);
       a=AA.data(:,2);
       b=AA.data(:,3);
end
d=round(H*sum(p));
D(1:NOJ)=d;
sched_memory=zeros(NOJ-1,NOJ);
% ==================initial tabu search =================
TabuSize=7;
itt_max=200;
Can_Sch = NOJ;
Max_rep =20;
TL=zeros(TabuSize,2);
objFunction = @(m)heu2(NOJ, H, p, a, b, m);
sched_curr = Initial_Sequence(NOJ, H, p, a, b);
ETCost_curr = objFunction(sched_curr);
Ini_Sched = sched_curr;
Ini_ETCost = ETCost_curr;
sched_best = sched_curr;
ETCost_best = ETCost_curr;
% =============== Tabu Search initial End
rep = 0;
itt=1;
Start_time = cputime;
while itt<=itt_max
% ======= Generate a set of trial solution ======
for g=1:Can_Sch
       [sched_new r s] = swapping(sched_curr);
       TS_memory(g,:) = [r s];
       sched_memory(g,:)=sched_new;
       ETCost_new(g)=objFunction(sched_new);
end
   TS_memory;
   sched_memory;
  ETCost_new_list=ETCost_new;
   ETCost_new=min(ETCost_new);
   ETCost_min_index=find(ETCost_new_list==ETCost_new);
   ETCost_length=length(ETCost_min_index);
   sched_memory_min=sched_memory(ETCost_min_index,:);
   ETCost_min_index=min(ETCost_min_index);
sched_new=sched_memory(ETCost_min_index,:);
   TS_new = sort (TS_memory(ETCost_min_index,:),'ascend');
% =========== Generate a set of trial solution End
% ================== Tabu List Check       xyz=0;
for xy=1:TabuSize
          xx=1;
          init=0;
while xx <= 2
if TS_new(1,xx)==TL(xy,xx)
                init=init+1;
end
            xx=xx+1;
end
if init==2
           xyz=1;
end
end
% ================ Tabu List Check end
% ========= Update Optimal Solution and Tabu List
if ETCost_new < ETCost_best
if xyz==0
         ETCost_best = ETCost_new;
         sched_best = sched_new;
         ETCost_curr = ETCost_new;
         sched_curr = sched_new;
         [Zero_As As] = min (TL(:,1));
if Zero_As == 0
           yz = As;
else
if itt < TabuSize
               yz=itt;
elseif mod(itt,TabuSize)==0
               yz = TabuSize;
else
              yz = mod(itt,TabuSize);
end
end
         TL(yz,:)= TS_new;
else
% ==================== Aspiration Criterian                  ETCost_best = 
ETCost_new;
         sched_best = sched_new;
         ETCost_curr = ETCost_new;
         sched_curr = sched_new;
for xy=1:TabuSize
           xx=1;
           init=0;
if TS_new(1,1)==TL(xy,1)&&TS_new(1,2)==TL(xy,2)
               TL(xy,:) = 0;
end
end
end
% ==================== Aspiration Criterian end else
if ETCost_new == ETCost_best
         rep = rep + 1;
end
if xyz==0
         ETCost_curr = ETCost_new;
         sched_curr = sched_new;
         [Zero_As As] = min (TL(:,1));
if Zero_As == 0
          yz = As;
else
if itt < TabuSize
               yz=itt;
elseif mod(itt,TabuSize)==0
               yz = TabuSize;
else
              yz = mod(itt,TabuSize);
end
end
         TL(yz,:)= TS_new;
end
end
% ========= Update Optimal Solution and Tabu List end
% ================== Stopping Criteria
if rep == Max_rep
      itt = itt_max + 1;
else
itt=itt+1;
end
% ================== Stopping Criteria end end
Stop_time = cputime - Start_time;
% ================== Tabu Search end
% ================== Results Displayed====
```

```
fprintf('----------------- Result --------------------\n');
disp (['Initial Solution ETCost = ', num2str(Ini_ETCost)]);
fprintf('The Best SolutionETCost is = %d\n',ETCost_best);
fprintf('The Calculation time is = %f\n',Stop_time);
fprintf('---------------------------------------------\n');
%=====================================================%**
```

Initial Sequence Function

```
%Initial Sequence Function%
function sched=Initial_Sequence(NOJ, H, p, a, b)
d=round(H*sum(p));
ej=p./a; % divided the processing time over earliness penalety
tj=p./b; % divided the processing time over earliness penalety
[R1,I1]=sort(ej,'descend');% order the early job in non increasing order
[R2,I2]=sort(tj,'ascend');% order the tardy job in nondecreasing order
pn1=p(I1);% the processing time accoeding to the early jop order
pn2=p(I2);% the processing time accoeding to the tardy jop order
S(1)=pn1(I1(1));% candiate scheduale for early
for i=2:NOJ
  SS(i)=sum(pn1([1:i]));
end
for i=1:NOJ
if (SS(i)<=d)
      SE(i)=I1(i);
end
end
 L=length(SE);
 S=[SE zeros(1,NOJ-L)];
 j=1;
for i=1:NOJ
if(I2(i)~=SE(1:L))
      S(L+j)=I2(i);
      j=j+1;
end
end
 sched = S; % the initial scheduled
%****************************************************%
```

Function For calculating Objective Function

```
%Herustic Function to Calculate The Objective Function Total Weighted
Earliness & tardiness penalty%
function f=heu2(NOJ, H, p, a, b, sched)
pn=p(sched);% the process time acording to the initial scheduled
an=a(sched);% the earliness penalty acording to the initial scheduled
bn=b(sched);% the tardiness penalty acording to the initial scheduled
d=round(H*sum(p));% common due date
D(1:NOJ)=d; % make the due date for all column of due date equality
C(1)=pn(1); % the complition time of the first jop
for i=2:NOJ
   C(i)= sum(pn([1:i])); % compute the completion time for every jop
end
Lj=C-D; % clalculate the lateness
for i=1:NOJ
   Ej(i)=max(0,-Lj(i));% clalculate the earlines
   Tj(i)=max(0,Lj(i));% clalculate the tardiness
end
f=sum(an'.*Ej)+sum(bn'.*Tj);%summation of all tardiness andmearliness
multiply by their penalty
%*************%
```

Swapping Function

```
%Swapping Function%
function [sched r s]=swapping(m)
NOJ=length(m);
a = 0;
while a == 0
   r=randi(NOJ);
   s=randi(NOJ);
if r~=s
      m(:,[r s]) = m(:,[s r]);
    a = 1;
end
end
sched=m;
%---------------------------------------------------%
```