ISSN NO. 2456-3129

International Journal of Engineering, Pure and Applied Sciences,
Vol. 1, No.2, 2016

Optimal Production Schedule with Arbitrary Lags and Job Block Criteria Consisting Graphical User Interface Using Matlab

Dr. Deepak Gupta[1], Harminder Singh[1] , Dr.Avdhesh kumar[2]
Department of Mathematics,Maharishi Markandeshwar University[1],Mullana(Ambala)India[1]
Department of applied sciences[2],RIMT Institute of Engineering & Technology, Mandi Gobindgarh[2]
Email:harminder.cheema85@gmail.com[1]

Abstract: This paper is an attempt to study n jobs over two machines where processing times are associated with respective probabilities and transportation time in addition with arbitrary lags i.e. start lag or stop lag. The objective of the study is to develop an efficient algorithm to compute the production rate in an effective manner or to minimize the total elapsed time in two stage flow-shop scheduling with different system parameters like transportation, arbitrary lags also job block criteria. A numerical illustration as well as GUI is design to clarify the proposed concept..

Index Terms-Flow-shop, start lag, stop lag, transportation time, job block, Graphical user interface.

1. INTRODCUTION

Schedulingis generally described as an allocation of a set of resources over time to perform a set of tasks. Scheduling emerges in various domains such as hospital management, airlines, trains, production scheduling etc. At each stage there is a machine to perform the required set of jobs. In other words scheduling refers to placing jobs in a certain order or sequence so as to minimize the elapsed time with no passing between the jobs. By the time lag we meant the minimum time delay which is required between the executions of two consecutive operations of the same job. In actual time lag represents the time when one job is moved one machine to another machine is negligible. The start lag (Di>,0) is the minimum time which must elapse between starting job i on the first machine and starting it on the second machine. The stop lag (Ei>,0) for the job i is the minimum time which elapsed between the completing it on second machine. Equivalent job-block concept in the theory of scheduling has many applications in the production concern, hospital management etc. where priority of one job over other becomes significant it may arise the additional cost for providing this facility.

2. LITERATURE SURVEY

Johnson [1] gave procedure for finding the optimal schedule for n-jobs, two machine flow-shop problem with minimization of the make span (i.e. total elapsed time) as the objective. Also Mitten and Johnson [2] combined discussed the n job,2 machines flow-shop Scheduling problem in which despite of processing times some additional tags are introduced. Maggu and Das [4]introduced the equivalent job-block concept in the theory of scheduling which has many applications in the production concern, hospital management etc. where priority of one job over other becomes significant it may arise the additional cost for providing this facility.Bagga [3], Maggu and Das [6],[7], Szwarch [8], Yoshida & Hitomi [5], Singh, Anup [13], etc. derived the optimal algorithm for two/ three or multistage flow shop problems taking into account the various constraints and criteria. Kern,W. ,Nawjin,W.M[10], Dell` Amico, M[11], J.Riezebos,G.J.C Goalman[12] continues with dealing different scheduling problems including time lags. Singh, T.P.and Gupta, D.[9],[13][16], associated probabilities with processing time and set up time,transportation time as well as concept of breakdown interval in their studies. Later, Singh, T.P, Gupta, D[14],[15],[17] studied two /multiple flow shop problem to minimize rental cost under a pre-defined rental policy in which the probabilities have been associated with processing time oneach machine. The present paper addresses the flowshop scheduling problem in which processing times are associated with probabilities with arbitrary lags, transportation and job block for effective scheduling.

Equivalent job block: Let there be two jobs i and j is a sequence S to be processed on two machines A and B in the order A B. Let the

ISSN NO. 2456-3129

International Journal of Engineering, Pure and Applied Sciences,
Vol. 1, No.2, 2016

equivalent job of i and j by denoted by a' Then
$A_a = A_i + A_j - \min(A_j, B_i)$
$B_a = B_i + B_j - \min(A_j, B_i)$
Where A_a and B_a denote the processing time of equivalent job 'a' on machine A and B respectively.

Total Elapsed Time: it is the time interval between starting the first job and completing the last job including the idle time (if any) in a particular order by the given set of the machines.
Processing Time (t_j): It is the time required to process job j. It includes both actual time as well as set-up time.

Effective transportation: The effective transportation time of job I denoted by $T^{`}_{i,s\to s+1}$ is defined as $T^{`}_{i,s\to s+1}$ =max(D_i-G_i ,E_i-H_i ,$T_{i,s\to s+1}$) where s=1,2,3,4….m-1
Where $G_i=A_{i1}+A_{i2}+A_{i3+}\ldots\ldots A_{i(m-1)}$ and $H_i= A_{i2}+A_{i3+}\ldots\ldots A_{im}$

Concept of Transportation time in flow shop:
In many practical situations of scheduling it is seen that machines are distantly situated and therefore, definite finite time is taken in transporting the job from one machine to another if the form of.

i) Loading time of jobs.
ii) Moving time of jobs.
iii) Unloading time of jobs.

The sum of all the above times has been designated by various researches as transportation time of a job. This transportation time is the amount of time required to dispatch the job i after it has been completed on machine A, to the next succeeding machine B for its onward processing. It is denoted by t_i for job i.

Assumptions

1. Each machine is assumed to be continuously available for the assignment of jobs.
2. No significant division of time scale into shifts or days for the machines is assumed.
3. No temporary availability of machines is assumed to meet the certain causes of the machines due to their break down or maintenance etc.
4. No partition of a job is assumed to be allowable.
5. No like machine of the same type is allowed.
6. Each machine can handle at most one operation at a time.
7. Pre-emption is not allowed, that is, once a job is started it is performed to completion.
8. The time intervals for processing are independent of the order in which the jobs are done.
9. The technological ordering of the machines operating the jobs is known to be predetermined.
10. The transportation times of jobs from one machine to the other is assumed to be negligible.

3. EQUIVALENT-JOB FOR A JOB-BLOCK THEOREM DUE TO MAGGU AND DAS (1977) IN TWO MACHINE FLOW-SHOP PROBLEM:

In processing a schedule S = (α_1, α_2, … α_{k-1}, α_k, α_{k+1}, α_{k+2}, …, α_n) of n-jobs on two machines A and B in the order AB with no passing allowed. The job-block (α_k, α_{k-1}) having processing times {$A_{\alpha k}$, $B_{\alpha k}$,$A_{\alpha k+1}$,$B_{\alpha k+1}$} is equivalent to the single job β (called equivalent-job β).

Now the processing times of job β on the machines A and B denoted respectively by A_β, B_β are given by

$A_\beta = A_{\alpha k} + A_{\alpha k+1} - \min\{B_{\alpha k}, A_{\alpha k+1}\}$,
$B_\beta = B_{\alpha k} + B_{\alpha k+1} - \min\{B_{\alpha k}, A_{\alpha k+1}\}$,

Proof: Let T_{pq} denote the completion time of job p on machine q for the given sequence S, We can consider the following relations:

$T_{\alpha k\,B} = \max\{T_{\alpha k\,A}, T_{\alpha k-1\,B}\} + B_{\alpha k}$
$= \max\{T_{\alpha k\,A} + B_{\alpha k}, T_{\alpha k-1\,B} + B_{\alpha k}\}$
$T_{\alpha k+1\,B} = \max\{T_{\alpha k+1\,A}, T_{\alpha k\,B}\} + B_{\alpha k+1}$
$= \max\{T_{\alpha k+1\,A}, T_{\alpha k\,A} + B_{\alpha k}, T_{\alpha k-1\,B} + B_{\alpha k}\} + B_{\alpha k+1}$ $= \max\{T_{\alpha k+1\,A} + B_{\alpha k+1}, T_{\alpha k\,A} + B_{\alpha k} + B_{\alpha k+1}, T_{\alpha k-1\,B} + B_{\alpha k} + B_{\alpha k+1}\}$
Now $T_{\alpha k+1\,A} = T_{\alpha k\,A} + A_{\alpha k+1}$
We have
$T_{\alpha k+1\,B} = \max\{T_{\alpha k\,A} + A_{\alpha k+1}, B_{\alpha k+1}, T_{\alpha k\,A} + B_{\alpha k} + B_{\alpha k+1}, T_{\alpha k-1\,B} + B_{\alpha k} + B_{\alpha k+1}\}$,
$T_{\alpha k+2\,B} = \max\{T_{\alpha k+2\,A}, T_{\alpha k+1\,B}\} + B_{\alpha k+2}$,
$= \max\{T_{\alpha k+2\,A}, T_{\alpha k\,A} + A_{\alpha k+1} + B_{\alpha k+1}, T_{\alpha k\,A} + B_{\alpha k} + B_{\alpha k+1}, T_{\alpha k-1\,B} + B_{\alpha k} + B_{\alpha k+1}\} + B_{\alpha k+2}$,
Now, it is obvious that
$T_{\alpha k+2\,A} = T_{\alpha k\,A} + A_{\alpha k+1} + A_{\alpha k+2}$,
Hence,
$T_{\alpha k+2\,B} = \max\{T_{\alpha k\,A} + A_{\alpha k+1} + A_{\alpha k+1}, T_{\alpha k\,A} + A_{\alpha k+1} + B_{\alpha k+1}, T_{\alpha k\,A} + B_{\alpha k} + B_{\alpha k+1}, T_{\alpha k-1\,B} + B_{\alpha k} + B_{\alpha k+1}\} + B_{\alpha k+2}$,
Since $\max\{T_{\alpha k\,A} + A_{\alpha k+1} + B_{\alpha k+1}, T_{\alpha k} + B_{\alpha k} + B_{\alpha k+1}\}$
$= T_{\alpha k\,A} + \max\{A_{\alpha k+1}, B_{\alpha k}\} + B_{\alpha k+1}$
Therefore, we have :
$T_{\alpha k+2\,B} = \max\{T_{\alpha k\,A} + A_{\alpha k+1} + A_{\alpha k+2}, T_{\alpha k\,A} + \max\{A_{\alpha k+1}, B_{\alpha k}\} + B_{\alpha k+1}, T_{\alpha k-1\,B} + B_{\alpha k} + B_{\alpha k+1}\} + B_{\alpha k+2}$ (1)

$T_{\alpha k+2\ A} = T_{\alpha k-1\ A} + A_{\alpha k} + A_{\alpha k+1} + A_{\alpha k+2}$,
$= T_{\alpha k\ A} + A_{\alpha k+1} + A_{\alpha k+2}$ (2)

Now define a sequence S′ as

$S' = (\alpha_1, \alpha_2, \dots \alpha_{k-1}, \alpha_k, \alpha_{k-1}, \beta, \alpha_{k+2}, \dots, \alpha_n)$,

Where, $A_\beta = A_{\alpha k} + A_{\alpha k+1} - C$, (3)

$B_\beta = B_{\alpha k} + B_{\alpha k+1} - C$, (4)

In (3) or (4) C is a constant.

Let T'_{pq} denote the completion time of job p on machine q in the sequence S′, so that

$T'_{\beta B} = \max\{T'_{\beta A}, T_{\alpha k-1\ B}\} + B_\beta$
$= \max\{T'_{\beta A} + B_\beta, T'_{\alpha k-1\ B} + B_\beta\}$,

$T'_{\alpha k+2\ B} = \max\{T'_{\alpha k+2\ A}, T'_{\beta A} + B_\beta\} + B_{\alpha k+2}$
$= \max\{T'_{\alpha k+2\ A}, T'_{\beta A} + B_\beta, T'_{\alpha k-1\ B} + B_\beta\} + B_{\alpha k+2}$ … (5)

Now, it is obvious that

$T'_{\alpha k+2\ B} = T'_{\alpha k-1\ A} + A_\beta + A_{\alpha k+2}$
$= T_{\alpha k-1\ A} + A_{\alpha k} + B_{\alpha k+1} - C + A_{\alpha k+2}$
(As $T'_{\alpha k-1\ A} = T_{\alpha k-1\ A}$) (6)
$= T_{\alpha k\ A} + A_{\alpha k+1} - C + A_{\alpha k+2}$
(As $T_{\alpha k\ A} = T_{\alpha k-1\ A} + A_{\alpha k}$)

$T'_{\beta A} = T'_{\alpha k-1\ A} + A_\beta$
$= T_{\alpha k-1\ A} + A_{\alpha k} + A_{\alpha k+1} - C$ (7)

$T'_{\beta A} = T_{\alpha k\ A} + A_{\alpha k+1} - C$

Using (3), (4), (5), (6) and (7), we have

$T'_{\alpha k+2\ B} = \max\{T_{\alpha k\ A} + A_{\alpha k+1} - C + A_{\alpha k+2}$,
$T_{\alpha k\ A} + A_{\alpha k+1} - C + B_{\alpha k} + B_{\alpha k+1} - C$,
$T'_{\alpha k-1\ B} + B_{\alpha k} + B_{\alpha k+1} - C\} + B_{\alpha k+2}$ (8)

Let $C = \min\{A_{\alpha k+1}, B_{\alpha k}\}$ (9)

Then $A_{\alpha k+1} - C + B_{\alpha k} = A_{\alpha k+1} - \min\{A_{\alpha k+1}, B_{\alpha k}\} + B_{\alpha k}$
$= \max\{A_{\alpha k+1}, B_{\alpha k}\}$ (10)

Also $T'_{\alpha k-1\ B} = T_{\alpha k-1\ B}$ (11)

Hence from (8), (9), (10) & (11), we have

$T'_{\alpha k+2\ B} = \max\{T_{\alpha k\ A} + A_{\alpha k+1} + A_{\alpha k+2} - C, T_{\alpha k\ A} + \max(A_{\alpha k+1}, B_{\alpha k}) + B_{\alpha k+1} - C, T_{\alpha k-1\ B} + B_{\alpha k} + B_{\alpha k+1} - C\} + B_{\alpha k+2}$
$= \max\{T_{\alpha k\ A} + A_{\alpha k+1} + A_{\alpha k+2}, T_{\alpha k\ A} + \max(A_{\alpha k+1}, B_{\alpha k}) + B_{\alpha k+1}, T_{\alpha k-1\ B} + B_{\alpha k} + B_{\alpha k+1}\} + B_{\alpha k+2} - C$ (12)

Hence from (1) and (12), we have

$T'_{\alpha k+2\ B} = T_{\alpha k+2\ B - C.}$ (13)

From (2) & (6), it is obvious that

$T'_{\alpha k+2\ A} = T_{\alpha k+2\ A - C.}$ (14)

From equations (13) and (14), it is clear that replacement of job-block (α_k, α_{k+1}) in S by job β decreases the completion times on both the machines of the later job α_{k+2} by a constant C in S′ as compared for the job : α_{k+2} in S. Let T and T′ be the completion times of sequences S and S′, respectively. Then from the above discussion, it is observed that T′ = T – C, hence when β replaces job α_k, α_{k+1} in any sequence s to produce a new sequence S′, the completion times on all the machines are changed by a value which is independent of the particular sequence S. Hence the substitution does not change the relative merit of different sequences. Hence, job β is equivalent job for job-block (α_k, α_{k+1}).

Theorem: Two-machine, n-job problem' with transportation times from our given original problem replacing three times (Start-lag, Stop-lag, transportation time) by single time t'_i

Proof: Let U_{ix} and T_{ix} denote Starting and Completion times of any job i on machine X (X = A, B, i = 1, 2, 3, ……, n) respectively in a sequence S. From definition of Start-lag D_i, we have

$U_{iB} - U_{iA} \geq D_i$

Now $T_{iA} - U_{iA} + A_i$

i.e.,Hence, we have, $U_{iB} - (T_{iA} - A_i) \geq D_i$

i.e.,$U_{iB} - T_{iA} \geq D_i - A_i$ (1)

From definition of Stop-lag E_i,

we have ,$T_{iB} - T_{iA} \geq E_i$,

Now , $T_{iB} - U_{iB} + B_i$

Hence, we have $U_{iB} + B_i - T_{iA} \geq E_i$

i.e., $U_{iB} - T_{iA} \geq E_i - B_i$ (2)

Also, from the definition of transportation time t_i, we have

$U_{iB} - T_{iA} \geq t_i$ (3)

Let $t'_i = \max\{D_i - A_i, E_i - B_i, t_i\}$ (4)

From (1), (2) and (3), it is obvious that

$U_{iB} - T_{iA} \geq t'_i$ … (5)

Remarks

(1) For every job I, $D_i = A_i$ and $E_i = B_i$then the algorithm reduces to Maggu and Das (1980) problem algorithm.

(2) If either $t_i = 0$, or $D_i \geq A_i + t_i, + B_i$, then the algorithm reduces to the Mitten-Johnson's (2) problem.

(3) If $t_i = 0$, $D_i = A_i$, $E_i = B_i$, then the algorithm reduces to Bellman's (6) and Johnson's [1] problem.

4. NOTATIONS

S: Sequence of job 1, 2, 3, ……., n

M_j : Machine j, j= 1, 2, ……….

A_i: Processing time of i^{th} job on machine A.

B_i: Processing time of i^{th} job on machine B.

A_α: Expected processing time of i^{th} job on machine A.

B_α: Expected processing time of i^{th} job on machine B.

p_i: Probability associated to the processing time A_i of i^{th} job on machine A.

q_i: Probability associated to the processing time B_i of i^{th} job on machine B.

β: Equivalent job for job-block.

S_i: Sequence obtained from Johnson's procedure to minimize rental cost.

D_i: Start lag for job i

E_i: Stop lag for job i

U_{ix}: Starting time of any job I on machine x

ISSN NO. 2456-3129

International Journal of Engineering, Pure and Applied Sciences,
Vol. 1, No.2, 2016

T_{ix}: Completion time of any job on machine x

$T_{i,j\to k}$: Transportation time of i^{th} job from j^{th} machine to k^{th} machine

$T^{`}_{i,j\to k}$:Effective transportation time from i^{th} job from j^{th} machine to k^{th} machine

CT(S): Completion time of 1^{st} job of each sequence S_i on machine A

Algorithm:

Step 1: : Define expected processing time A_α as shown in table 1

Step 2: let $t_i^{`}$ denote the effective transportation times, defined by $t_i^{`}$= max (D_i-A_{i} , E_i-B_{i} ,t)

Step 3: Define two fictitious machines G & H with processing time G_i& H_i for job I on G & H respectively, defined as:

$G_i= A_{i+} t_i^{`}$ and $H_i = b_i+t_i^{`}$

Step4: Take equivalent job α = (i_k, i_m) for the given job block (i_k, i_m) and define its processing time using below:

$$G_\alpha = G_k + G_m - \min(G_m, H_k)$$
$$H_\alpha = H_k + H_m - \min(G_m, H_k)$$

Step5: Apply Johnson's (1954) technique to obtain the optimal string Si for the new reduced problem obtained in step 4.

Step6: obtain in –out table for given problem in order to find out the total elapsed time.

```
#define Max 10
// *************** variable declaration
***************//
static float jobs[Max][7]; //to store jobs processing
time, probability, Transportation Time, StartLag,
StopLag
static float exp_pro_time[Max][2];// to store Expected
Processing Time for Both Machine
static float exp_trans_time[Max];// to store Expected
Transportation Time
static float fictitious_time[Max][2];// to store
Factitious Time
static float job_block_table[Max][2];
static float in_out[Max][4];
int total_jobs; // to store total number of jobs
int job_block[2]; // to store job block
char johnson[Max+1];
// ************** end variable declaration
**************//
void get_job_detail() //function to get jobs Processing
time and probabiity for both machine and
Transporation Time , StartLag and StopLag
{int
i,j;/*floatarr2[5][7]={24,.3,10.0,.2,4.0,10.0,12.0,16.0,.
2,9.0,.3,13.0,7.0,10.0,
22.0,.2,8.0,.2,9.0,6.0,5.0,28.0,.1,12.0,.1,2.0,4.0,4.0,
20.0,.2,13.0,.2,5.0,8.0,7.0 };
 total_jobs=5;for(i=0;i<total_jobs;i++) {
for(j=0;j<7;j++) { jobs[i][j]=arr2[i][j]; }
}*/printf("\nEnter Number of Job for Two Machines :
");scanf("%d",&total_jobs);for( i=0;i<total_jobs;i++
){printf("\n\n Enter The Detail for The Job [ %d
]",i+1);printf("\n\nEnter Processing Time  for
Machine 1 : ");scanf("%f",&jobs[i][0]);// geting
Processing Time for the Job;printf("Enter Probility for
Machine 1       : ");scanf("%f",&jobs[i][1]);// geting
Probility for the jobprintf("Enter Processing Time  for
Machine 2 : ");scanf("%f",&jobs[i][2]);// geting
Processing Time for the Job;printf("Enter Probility for
Machine 2       : ");scanf("%f",&jobs[i][3]);// geting
Probility for the jobprintf("Enter Transportation Time
:");
scanf("%f",&jobs[i][4]);//geting Transportation Time
for job
printf("Enter Start Lag  : ");
scanf("%f",&jobs[i][5]);// geting Start Lag
printf("Enter Stop Lag  : ");
scanf("%f",&jobs[i][6]);// geting Stop
Lag}//*/if(total_jobs>1){printf("\n\n Enter 1st Job no
for Job Block : ");scanf("%d",&job_block[0]);
printf("\n\n Enter 2nd Job no for Job Block :
");scanf("%d",&job_block[1]);
}}// end of get_job_detail function//void
put_job_detail()  //function to show jobs Processing
time , probabiity, Transportation time , startLag,
stopLag for machine
{int i,j;printf("\n\t  Machine1\t  Machine2\t \n");
printf("\nJobs\tTime\tProb.\tTime\tProb.\tTr.Time\tSt
Lag\tSpLag");
for(i=0;i<total_jobs;i++){printf("\n\n[ %d
]",i+1);printf("\t%.1f",jobs[i][0]);
printf("\t%.1f",jobs[i][1]);printf("\t%.1f",jobs[i][2]);
printf("\t%.1f",jobs[i][3]);printf("\t%.1f",jobs[i][4]);pr
intf("\t%.1f",jobs[i][5]);
printf("\t%.1f",jobs[i][6]);}}// end of Put_job_detail
function//
void get_exp_pro_time() // function to calculatig
expected processing time
{int
i,j;for(i=0;i<total_jobs;i++){exp_pro_time[i][0]=jobs[
i][0]*jobs[i][1];
exp_pro_time[i][1]=jobs[i][2]*jobs[i][3];}}// end of
get_exp_pro_time function//
void put_exp_pro_time() // function to showing
expected processing time
{int i,j;printf("\n\nExpected Proessing Time : -
");printf("\n\nJobs\t  Machine1\t  Machine2
\n");for(i=0;i<total_jobs;i++){
printf("\n\n[ %d
]",i+1);for(j=0;j<2;j++){printf("\t\t%.1f",exp_pro_tim
e[i][j]);}
}}// end of Put_exp_pro_time function//float
get_max(float a,float b,float c)
{if(a>b && a>c)return a;else if (b>a && b>c)return
b;elsereturn c}
void get_exp_trans_time() // function is used to get
expected transportation time
{int
i;for(i=0;i<total_jobs;i++){exp_trans_time[i]=get_ma
x(jobs[i][5]-exp_pro_time[i][0],jobs[i][6]-
```

```
exp_pro_time[i][1],jobs[i][4]);}}//end of
get_exp_trans_time()void put_exp_trans_time(){int
i;printf("\n\n Expected Transportation Time :-
\n");printf("\nJobs\tTrans.Time\n");for(i=0;i<total_job
s;i++)
{ printf("\n%d\t%3.1f",i+1,exp_trans_time[i]);}}//
end of put_trans_time functionvoid
get_fictitious_time(){int i;for(i=0;i<total_jobs;i++)
{fictitious_time[i][0]=exp_pro_time[i][0]+exp_trans_
time[i];fictitious_time[i][1]=exp_pro_time[i][1]+exp_
trans_time[i];}}// end of function get_factitious_time
void put_fictitious_time(){int i,j;printf("\n\n Fictitious
Time :-\n");printf("\nJobs\tMachine A\tMachine
B\n");for(i=0;i<total_jobs;i++)
{printf("\n%d",i+1);for(j=0;j<2;j++)
printf("\t%.1f\t",fictitious_time[i][j]);}}
// end of function put_factitious_timefloat Min(float
a,float b)
{if(a>b)return b;elsereturn a;}void
get_job_block_table(){
int
i,ii=0,flag=0;for(i=0;i<total_jobs;i++){if(job_block[0]
-1==i || job_block[1]-1==i) {if(job_block[0]-1==i)
{job_block_table[ii][0]=fictitious_time[i][0]+fictitiou
s_time[job_block[1]-1][0]-
Min(fictitious_time[job_block[1]-
1][0],fictitious_time[i][1]);job_block_table[ii][1]=ficti
tious_time[i][1]+fictitious_time[job_block[1]-1][1]-
Min(fictitious_time[job_block[1]-
1][0],fictitious_time[i][1]);ii++; }
}els{job_block_table[ii][0]=fictitious_time[i][0];
job_block_table[ii][1]=fictitious_time[i][1]; ii++;
}}}//============================ end of
get_job_block_table funciton
void put_job_block_table()
{int i,j;printf("\n Jobs\tGi\tHi\n");for(i=0;i<total_jobs-
1;i++){if(i==job_block[0]-1)printf("\n%c",225);else
if(i>=job_block[1]-1)printf("\n%d",i+2);
elseprintf("\n%d",i+1);for(j=0;j<2;j++){printf("\t%.1f
",job_block_table[i][j]);}
}}// end of put_job_block_table funciton
void get_johnson_rule(){int
i,j,k,jj,kk,strl=0,lb=0,rb=0,flag=0,f;
float min;char str[50];rb=total_jobs-
1;for(i=0;i<total_jobs-
1;i++){min=9999;for(j=0;j<total_jobs-1;j++)
{for(f=0;f<strl;f++){if(j==str[f]-
48)break;}if(f!=strl)continue;for(k=0;k<2;k++)
{if(min>=job_block_table[j][k]){min=job_block_tabl
e[j][k];jj=j;kk=k;}}
}//end of jif(jj>=job_block[1]-
1)flag=1;elseflag=0;if(kk==0){
if(jj==job_block[0]-
1){johnson[lb]=jj+flag+48;lb++;str[strl]=jj+48;
strl++;johnson[lb]=job_block[1]+48+flag;lb++;}else{
johnson[rb]=jj+48+flag;
rb--;str[strl]=jj+48;strl++;}}// end of k
else{if(jj==job_block[0]-1){
johnson[rb]=job_block[1]+47+flag;rb--
;str[strl]=jj+48;strl++;johnson[rb]=jj+48+flag;rb--
;}else{johnson[rb]=jj+48+flag;
rb--
;str[strl]=jj+48;strl++;}}johnson[total_jobs]=NULL;p
rintf("\n Jonson rule : %s",johnson);
}// end of get_johnson_rule
=======================================
================
void put_johnson_rule(){int i;i=0;printf("\n\n Jonson
Rule :-
\n\n");while(johnson[i]!=NULL){printf("\t%c",johnso
n[i]+1);i++;}
}// end of put_johnson_rule function
=======================================
=======
void get_in_out()
{int i;int job;float prev1=0.0;float
prev2=0.0;for(i=0;i<total_jobs;i++)
{ job=johnson[i]-48; in_out[i][0]=prev1; // machine A
starting Time
        in_out[i][1]=prev1+fictitious_time[job][0];
// machine A ending time
        prev1=in_out[i][1];
in_out[i][2]=get_max(prev1+exp_trans_time[job],pre
v2,0); // machine B starting
Timein_out[i][3]=in_out[i][2]+fictitious_time[job][1];
// machine B ending time
        prev2=in_out[i][3];}}
// end of get_in_out function
=======================================
===============
void put_in_out()
{int i;printf("\n\nIn-Out Table :-\n\n");
printf("\nJobs\tMachine_A\tExpected\tMachine_B");
        printf("\n\tIn-Out\t\tTrans.Time\tIn-Out\n");
for(i=0;i<total_jobs;i++){printf("\n%d",johnson[i]-
47);printf("\t%.1f-
%.1f",in_out[i][0],in_out[i][1]);printf("\t%.1f\t",exp_t
rans_time[johnson[i]-48]);printf("\t%.1f-
%.1f",in_out[i][2],in_out[i][3]);}}
// end of put_in_out function
=======================================
===============
void show_me()
{int
gm=0,gd=0;initgraph(&gd,&gm,"");settextstyle(4,0,1
0);outtextxy(getmaxx()/2-200,getmaxy()/2-
100,"Welcome");setcolor(14);settextstyle(1,0,1);outte
xtxy(getmaxx()-350,getmaxy()-30,"Designed &
Developed by Harminder Singh");}void main(){
clrscr();show_me();getch();cleardevice();get_job_deta
il();put_job_detail();
getch();get_exp_pro_time();put_exp_pro_time()
;getch();get_exp_trans_time();put_exp_trans_time();g
etch();get_fictitious_time();
put_fictitious_time();getch();get_job_block_table();pu
t_job_block_table();
```

ISSN NO. 2456-3129

International Journal of Engineering, Pure and Applied Sciences,
Vol. 1, No.2, 2016

getch();get_johnson_rule();put_johnson_rule();getch()
;get_in_out();
put_in_out();getch();}

5. NUMERICAL ILLUSTRATION:

Obtain optimal sequence for 5 jobs and 2 machines problem given by the following tableau1:

JOBS	Machine A		Machine B		Transportation Time $T_{i,j} \to k$	Start Lag D_i	Stop Lag E_i
	a_i^1	p_i	a_i^2	q_i			
1	24	.3	10	.2	4	10	12
2	16	.2	9	.3	13	7	10
3	22	.2	8	.2	9	6	5
4	28	.1	12	.1	2	4	4
5	20	.2	13	.2	5	8	7

Tableau 1

Our objective is to minimize the total rental cost of the machine, in which jobs are to be processed as a group job (2, 4)

Solution: As per step 1: Define expected processing time A_α, B_α on both machines A and B respectively as shown in tableau-2

JOBS	Machine A_α	Machine B_α	Transportation Time $T_{i,j} \to k$	Start Lag D_i	Stop lag E_i
1	7.2	2	4	10	12
2	3.2	2.7	13	7	10
3	4.4	1.6	9	6	5
4	2.8	1.2	2	4	4
5	4	2.6	5	8	7

Tableau 2

As per step 2: t_1^1= max (Di□Ai, Ei□Bi, ti)
=(10□7.2,12□2,4)
=max (2.8,10,4) =10
t_2^1= max(7-3.2,7-2.7,13)=13
t_3^1= max (6-4.4,5-1.6,9) =9
t_4^1= max(4-2.8,4-1.2,2) =2.8
t_5^1= max(8□4, 7□2.6, 5) =5
so by above we find the expected transportations time as make fictitious machines as shown in tableau 3
As per step 3

Jobs	G_i	H_i
1	7.2+10=17.2	2+10=12
2	3.2+13=16.2	2.7+13=15.7
3	4.4+9=13.4	1.6+9=10.6
4	2.8+2.8=5.6	1.2+2.8=4
5	4+5=9	2.6+5=7.6

Tableau 3

As per Step 4 Using equivalent job block criteria β over job (2,4)
G_α = 16.2+5.6-min(5.6,15.7) = 16.2
H_α= 15.7+7.6-min (5.6,15.7) = 17.7

Jobs	G_i	H_i
1	17.2	12
β	16.2	14.1
3	13.4	10.6
5	9	7.6

Tableau4

As per step 5**:** by applying Johnson rule:

5	β	1	3

Also

5	2	4	1	3

As per step 6**:** we prepare in our table as shown in tableau 6

Jobs	Machine A IN-OUT	Expected transpor-tation time $T^{`\square}_{j} \to k$	Machine B IN-OUT
5	0-9	5	14-21.6
2	9-25.2	13	38.2-53.9
4	25.2-30.8	2.8	53.9-57.9
1	30.8-68.3	10	78.3-90.3
3	68.3-81.7	9	90.7-101.3

Tableau 6

Hence CT(S):=Total elapsed time is 101.3 and optimal sequence S_i is 5,2,4,1,3

REFERENCES

[1] Johnson S. M. (1954), “optimal two and three stage production schedule with set up times included” Nay Res Log Quart Vol 1 pp 61-68.

[2] Mitten,L.G(1958)”sequencing n jobs on machines with arbitrary time lags” management sciences,29,477-481

[3] Bagga, P.C.(1969), "Sequencing in a rental situation", Journal of Candian Operation Research Society 7, pp.152-153.

[4] Maggu, P.L and Das, G, (1977), "Equivalent-job block in job scheduling" Operation research, Vol. 14, No.4 pp 277-281.

[5] Yoshida and Hitomi (1979), **'**Optimal two stage production scheduling with set up times separated**',**AIIETransactions. Vol. II.pp 261-263.

[6] Maggu, P.L and Dass, G, (1980), "nx2 scheduling with transportation time" PAMS vol (2) pp 1-6.

[7] Maggu & Das (1981), **'**On n x 2 sequencing problem with transportation time of jobs**',** Pure and Applied Mathematika Sciences, 12-16**.**

[8] Szwarc,W(1983), "Flow shop problems with time lags" Management Science29,477-481

[9] Singh , T.P., On n x 2 flow shop problem solving job block, Transportation times, Arbitrary time and Break-down machine times, PAMS Vol. XXI, No. 1-2 (1985).

[10] Kern,W.,Nawjin,W.M(1991) "Scheduling multi-operation jobs with time lags on a single machine" Working paper University of Twente, Holland.

[11] Dell` Amico, M (1993), "Shop problems with two machines and time lags"Rapport Inferno No.93/20,Poltecnico Di Milano, Dipatimento Di Ellectronica E Informazione,Italy.

[12] J.Riezebos,G.J.C Goalman, "Time lag sze in multiple operations flowshop scheduling heuristics"European journal of operation research 105(1998)72-90.

[13] Anup (2002),"On two machine flow shop problem in which processing time assume Probabilities and there exists equivalent for an ordered job block", JISSO, Vol XXIII No.1-4,pp.41-44.

[14] Singh T.P., K Rajindra and Gupta Deepak, (2005)" Optimal three stage production schedule the processing time and set up times associated with probabilities including job block" Proceedings of National Conference FACM pp.463-470.

[15] Singh, T.P, Gupta Deepak (2006), Minimizing rental cost in two stage flow shop , the processing time associated with probabilities including job block, Reflections de ERA, Vol 1. issue 2, pp 107-120.

[16] Singh, T.P. & KUMAR, R. & GUPTA, D. [2005] "Optimal three stage production schedule, the processing and set up times associated with probabilities including job block criteria" published in proceedings of National Conference on FACM, pp 463-470.

[17] Singh ,T.P., Vij, Indria and Kumar, sunil, (2007), "Optimal Rental Cost in Restrictive 3-Stage scheduling including Job Block concept" Acta Ciencia Indica, vol. XXXIII, no.3, pp. 761-767