Minimizing Rental Cost under Specified Rental Policy in Two Stage Flow Shop, the Processing Time Associated with Probabilities Including Break-down Interval and Job – Block Criteria

Sameer Sharma (Corresponding author)
Assistant Professor, Dept. of Mathematics,
D.A.V. College, Jalandhar, Punjab, India
Tel: 011+91-9814819064, Email: samsharma31@yahoo.com

Deepak Gupta,
Prof. & Head, Dept. of Mathematics,
Maharishi Markandeshwar University, Mullana, Ambala ,India
Tel: 011+91-9896068604, Email: guptadeepak2003@yahoo.co.in

Abstract
In real world scheduling applications, machines might not be available during certain time periods due to deterministic or stochastic causes. This paper is an attempt to study the two machine general flow shop problem in which the processing time of the jobs are associated with probabilities, following some restrictive renting policy including break-down interval and equivalent job-block criteria. The objective of the paper is to find an algorithm to minimize the rental cost of the machines under specified rental policy with break-down interval and job block criteria. The proposed method is very simple and easy to understand and also, provide an important tool for decision makers. The method is justified with the help of numerical example and a computer program.

Keywords: Equivalent-job, Rental Policy, Makespan, Elapsed time, Idle time, Break-down interval, Johnson's technique, Optimal sequence.

1. Introduction

The classical scheduling literature commonly assumes that the machines are never unavailable during the process. This assumption might be justified in some cases but it does not apply if certain maintenance requirements, break-downs or other constraints that causes the machine not to be available for processing have to be considered. The temporal lack of machine availability is known as '***break-down***'. Before 1954, the concept of break-down of machines had not considered by any author. In 1954 Johnson had considered the effect of break-down of machines on the completion times of jobs in an optimal sequence. Later on many researchers such as *Adiri* [1989], *Akturk and Gorgulu* [1999], *Smith* [1956], *Szwarc*[1983], *Chandramouli* [2005], *Singh T.P.* [1985] , *Belwal and Mittal* [2008] etc. have discussed the various concepts of break-down of machines. The functioning of machines for processing the jobs on them is assumed to be smooth with having no disturbance on the completion times of jobs. But there are feasible sequencing situations in flow shops where machines while processing the jobs get sudden break-down due to failure of a component of machines for a certain interval of time or the machines are supposed to stop their working for a certain interval of time due to some external imposed policy such as stop of flow of electric current to the machines may be a government policy due to shortage of electricity production. In each case this may be well observed that working of machines is not continuous and is subject to break for a certain interval of time.

In flow-shop scheduling, the object is to obtain a sequence of jobs which when processed in a fixed order of machines, will optimize some well defined criteria. Various Researchers have done a lot of work in this direction. Johnson [1954], *Ignall* and *Scharge* [1965], *Szwarch* [1977]. *Chandra Shekhran* [1992], *Maggu & Das* [1977], *Bagga P.C.* [1969], *Singh T.P., Gupta Deepak* [2005] etc. derived the optimal algorithm for two, three or multi stage flow shop problems taking into account the various constraints and criteria. *Maggu & Das* [1977] introduced the concept of equivalent-job blocking in the theory of scheduling. The concept is useful and significant in the sense to create a balance between the cost of providing priority in service to the customer and cost of giving services with non priority customers. The decision maker may decide how much to charge extra from the priority customer. Further, *Maggu* [1977], *Singh T.P* and *Gupta Deepak* [2005] associated probabilities with processing time and set up time in their studies. Later, *Singh T.P., Gupta Deepak* [2006] studied n x 2 general flow shop problem to minimize rental cost under a pre-defined rental policy in which the probabilities have been associated with processing time on each machine including job block criteria. We have extended the study made by *Singh T.P., Gupta Deepak* by introducing the concept of break-down interval. We have developed an algorithm minimizing the utilization time of second machine combined with Johnson's algorithm in order to minimize the rental cost of machines.

2. Practical Situation

Various practical situations occur in real life when one has got the assignments but does not have one's own machine or does not have enough money or does not want to take risk of investing huge amount of money to purchase machine. Under such circumstances, the machine has to be taken on rent in order to complete the assignments. In his starting career, we find a medical practitioner does not buy expensive machines say X-ray machine, the Ultra Sound Machine, Rotating Triple Head Single Positron Emission Computed Tomography Scanner, Patient Monitoring Equipment, and Laboratory Equipment etc., but instead takes on rent. Rental of medical equipment is an affordable and quick solution for hospitals, nursing homes, physicians, which are presently constrained by the availability of limited funds due to the recent global economic recession. Renting enables saving working capital, gives option for having the equipment, and allows upgradation to new technology.

Sometimes the priority of one job over the other is preferred. It may be because of urgency or demand of its relative importance, the job block criteria becomes important.

Another event which is mostly considered in the models is the break-down of machines. There may also be delays due to material, changes in release and tail dates, tools unavailability, failure of electric current, the shift pattern of the facility and fluctuations in processing times. All of these events complicate the scheduling problem in most cases. Hence the criterion of break-down interval becomes significant.

3. Notations

- S : Sequence of jobs 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_i' : Expected processing time of i^{th} job on machine A.
- B_i' : 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.

L : Length of the break-down interval.

$A_i^{''}$: Expected processing time of i^{th} job after break-down effect on machine A .

$B_i^{''}$: Expected processing time of i^{th} job after break-down effect on machine B.

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

C_j : Rental cost per unit time of machine j.

U_i : Utilization time of B (2^{nd} machine) for each sequence S_i

$t_1(S_i)$: Completion time of last job of sequence S_i on machine A.

$t_2(S_i)$: Completion time of last job of sequence S_i on machine B.

$R(S_i)$: Total rental cost for sequence S_i of all machines.

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

4. Assumptions

1. We assume the rental policy that all the machines are taken on rent as and when they are required and are returned as when they are no longer required for processing. Under this policy second machine is taken on rent at time when first job completes its processing on first machine. Therefore idle time of second machine for first job is zero.

2. Jobs are independent to each other.

3. Machine break-down interval is deterministic, .i.e. the break-down intervals are well known in advance. This simplifies the problem by ignoring the stochastic cases where the break-down interval is random.

4. Pre- emption is not allowed, .i.e. once a job started on a machine, the process on that machine can't be stopped unless the job is completed.

7. Definitions

7.1 Definition 1:

An operation is defined as a specific job on a particular machine.

7.2 Definition 2:

Sum of idle time of M_2 (for all jobs)

$$\sum_{i=1}^{n} I_{i2} = \max\left[\left(\sum_{i=1}^{n} A_i^{'} - \sum_{i=1}^{n-1} B_i^{'}\right), \left(\sum_{i=1}^{n-1} A_i^{'} - \sum_{i=1}^{n-2} B_i^{'}\right), \left(\sum_{i=1}^{n-2} A_i^{'} - \sum_{i=1}^{n-3} B_i^{'}\right), ---\left(\sum_{i=1}^{2} A_i^{'} - \sum_{i=1}^{2-1} B_i^{'}\right), A_i^{'}\right]$$

$$= \max\left[P_n, P_{n-1}, P_{n-2}, -----, P_2, P_1\right]$$

$$= \max_{i \le k \le n}\left[P_k\right], \text{ where } P_k = \left(\sum_{i=1}^{k} A_i^{'} - \sum_{i=1}^{k-1} B_i^{'}\right) P_k = \left(\sum_{i=1}^{k} A_i^{'} - \sum_{i=1}^{k-1} B_i^{'}\right)$$

7.3 Definition 3:

Total elapsed time for a given sequence.

= Sum of expected processing time on 2^{nd} machine (M_2) + Total idle time on M_2

$$= \sum_{i=1}^{n} B'_i + \sum_{i=1}^{n} I_{i2} = \sum_{i=1}^{n} B'_i + \max\left[P_k\right], \text{ where } P_K = \sum_{i=1}^{k} A'_i - \sum_{i=1}^{k-1} B'_i .$$

8. Theorem's

Theorem 8.1:

Equivalent job block theorem due to Maggu & Das [1977]. In two machine flow shop in processing a schedule S=(α_1, α_2,… α_{k-1}, α_k, α_{k+1}… α_n) of n jobs on two machines A & B in the order AB with no passing allowed the job block (α_k, α_m) having processing times $A_{\alpha k}$,$B_{\alpha k}$,$A_{\alpha m}$,$B_{\alpha m}$ is equivalent to the single job β(called equivalent job β). The processing times of equivalent job β on the machines A & B denoted respectively by A_β and B_β are given by

$A_\beta = A_{\alpha k}+A_{\alpha m} - \min(B_{\alpha k},A_{\alpha m})$

$B_\beta = B_{\alpha k}+B_{\alpha m} - \min(B_{\alpha k},A_{\alpha m})$

Theorem 8.2:

Job *i* precedes to job *j* in optimal ordering, having minimum idle time on B if

$$\min(A_i', B_j') \le \min(A_j', B_i')$$

where A_i' = Expected processing time of i^{th} job on A = $A_i \times p_i$

B_j' = Expected processing time of i^{th} job on B = $B_i \times q_i$

Proof**:**

Let two sequences S_1 and S_2 of n jobs differ with job j and j+1 (j≠1) interchange in their positions.

S_1 = *1 ,2, 3 ,4,…,j+1 ,j, j+1 ,j+2 ……,n.*

S_2 = *1, 2, 3, 4…..,j-1, j+1 ,j, j+2, …., n.*

By definition, $P_k' \,\&\, P_k''$ for sequences S_1 & S_2 respectively will be same for k = 2,3,…,j-1,j+2,…n.

i.e. $P_k' = P_k''$ for k = 2,3,…,j-1,j, j+2,…n.

Now only $P_j', P_{j+1}', P_j'', P_{j+1}''$ are left to be determined

$$P_j' = \sum_{i=1}^{j} A_i' - \sum_{i=1}^{j-1} B_i' \quad \text{…..(1)}$$

$$P_j'' = \sum_{i=1}^{j} A_i' - \sum_{i=1}^{j-1} B_i' + A_{j+1}' \quad \text{…..(2)}$$

$$P_{j+1}' = \sum_{i=1}^{j-1} A_i' - \sum_{i=1}^{j-1} B_i' \quad \text{…..(3)}$$

$$P_{j+1}'' = \sum_{i=1}^{j+1} A_i' - \sum_{i=1}^{j-1} B_i' - B_{j+1}' \quad \text{……(4)}$$

On subtracting (1) from (3), we get

$$P_{j+1}' - P_j' = A_{j+1}' - B_j' \quad \text{……(5)}$$

On subtracting (1) from (2), we get

$$P_j'' - P_j' = -A_j' + A_{j+1}'$$

$$\Rightarrow \quad P_j'' = P_j' - A_j' + A_{j+1}' \quad \text{…..(6)}$$

On subtracting (3) from (4), we get

$$P_{j+1}'' - P_{j+1}' = B_j' - B_{j+1}'$$

$$P_{j+1}'' = P_{j+1}' + B_j' - B_{j+1}' \quad \text{…..(7)}$$

Sequence S_1, will give min. idle time in comparison to S_2 if

$$\max(P'_j, P'_{j+1}) < \max(P''_j, P''_{j+1})$$

$$\Rightarrow \max(P'_j, P'_{j+1}) < \max(P'_j - A'_j + A'_{j+1}, P'_{j+1} + B'_j - B'_{j+1}) \qquad \text{(using (6) \&(7))}$$

On subtracting $(P'_j + A'_{j+1})$ from both sides, we get

$$\max(P'_j - P'_j - A'_{j+1}, P'_{j+1} - P'_j - A'_{j+1}) < \max(-A'_j, P'_{j+1} - P'_j - A'_{j+1} + B'_j - B'_{j+1})$$

$$\Rightarrow \max(-A'_{j+1}, P'_{j+1} - P'_j - A'_{j+1}) < \max(-A'_j, P'_{j+1} - P'_j - A'_{j+1} + B'_j - B'_{j+1})$$

$$\Rightarrow \max(-A'_{j+1}, A'_{j+1} - B'_j - A'_{j+1}) < \max(-A'_j, A'_{j+1} - B'_j - A'_{j+1} + B'_j - B'_{j+1}) \qquad \text{(using (5))}$$

$$\Rightarrow \max(-A'_{j+1}, -B'_j) < \max(-A'_j, -B'_{j+1})$$

$$\Rightarrow -\min(A'_{j+1}, B'_j) < -\min(A'_j, B'_{j+1})$$

$$\min(A'_{j+1}, B'_j) > \min(A'_j, B'_{j+1})$$

$$\min(A'_j, B'_{j+1}) < \min(A'_{j+1}, B'_j) \qquad \text{….. (8)}$$

Also $\min(A'_j, B'_{j+1}) = \min(A'_{j+1}, B'_j)$ ….. (9)

(if S_1 & S_2 are in-different)

From (8) & (9), we conclude that sequence S_1 will be preferable to S_2 if

$$\min(A'_j, B'_{j+1}) \le \min(A'_{j+1}, B'_j)$$

If these conditions hold then job j precedes over j+1 for optimal order having minimum idle time.

9. Algorithm

Based on the equivalent job block theorem by *Maggu & Das* and by considering the effect of break-down interval (a ,b) on different jobs, the algorithm which minimize the total rental cost of machines under specified rental policy with the minimum makespan can be depicted as below**:**

*Step 1***:** Define expected processing time A_i' & B_i' on machine A & B respectively as follows:

$A_i' = A_i \times p_i$

$B_i' = B_i \times q_i$

Step 2: Define expected processing time of job block β = (k ,m) on machine A & B using equivalent job block given by *Maggu & Das* i.e. find A_β' and B_β' as follows:

$A_\beta' = A_k' + A_m' - \min(B_k', A_m')$

$B_\beta' = B_k' + B_m' - \min(B_k', A_m')$

Step 3: Using *Johnson's* two machine algorithm obtain the sequence S, while minimize the total elapsed time.

Step 4: Prepare a flow time table for the sequence obtained in step 3 and read the effect of break-down interval (a ,b) on different jobs on the lines of *Singh T.P*. [1985].

Step 5: Form a reduced problem with processing times A_i'' and B_i''.

If the break-down interval (a, b) has effect on job *i* then

$A_i'' = A_i' + L$

$B_i'' = B_i' + L$; Where L = b – a, the length of break-down interval

If the break-down interval (a, b) has no effect on i^{th} job then

$A_i'' = A_i'$

$B_i'' = B_i'$

Step 6: Find the processing times A_β'' and B_β'' of job-block $\beta(k,m)$ on machine A and B using equivalent job-block β as in step 2.

Step 7: Now repeat the procedure to get the sequence S_i, using Johnson's two machine algorithms as in step 3.

Step 8: Observe the processing time of 1^{st} job of S_1 on the first machine A. Let it be α.

Step 9: Obtain all the jobs having processing time on A greater than α. Put these job one by one in the 1^{st} position of the sequence S_1 in the same order. Let these sequences be $S_2, S_3, S_4, \ldots\ldots, S_r$

Step 10: Prepare in-out flow table only for those sequence S_i $(i=1,2,\ldots r)$ which have job block β(k, m) and evaluate total completion time of last job of each sequence, .i.e. $t_1(S_i)$ & $t_2(S_i)$ on machine A & B respectively.

Step 11: Evaluate completion time $CT(S_i)$ of 1^{st} job of each of above selected sequence S_i on machine A.

Step 12: Calculate utilization time U_i of 2^{nd} machine for each of above selected sequence S_i as:

$U_i = t_2(S_i) - CT(S_i)$ for $i=1, 2, 3, \ldots r.$

Step 13: Find Min $\{U_i\}$, i=1, 2 …r. let it be corresponding to $i = m$, then S_m is the optimal sequence for minimum rental cost.

Min rental cost = $t_1(S_m) \times C_1 + U_m \times C_2$

Where C_1 & C_2 are the rental cost per unit time of 1^{st} & 2^{nd} machines respectively.

10. Programme

```
#include<iostream.h>
#include<stdio.h>
#include<conio.h>
#include<process.h>
void display();
void schedule(int,int);
void inout_times(int []);
void update();
void time_for_job_blocks();
float min;
int job_schedule[16];
int job_schedule_final[16];
int n;
float a1[16],b1[16];
float a1_jb,b1_jb;
float a1_temp[15],b1_temp[15];
int job_temp[15];
int group[2];//variables to store two job blocks
int bd1,bd2;//break down interval
float a1_t[16], b1_t[16];
float a1_in[16],a1_out[16];
float b1_in[16],b1_out[16];
float ta[16]={32767,32767,32767,32767,32767},tb[16]={32767,32767,32767,32767,32767};
void main()
```

```
{
	clrscr();
	int a[16],b[16];
	float p[16],q[16];
	int optimal_schedule_temp[16];
	int optimal_schedule[16];
	float cost_a,cost_b,cost;

	float min; //Variables to hold the processing times of the job blocks
	cout<<"How many Jobs (<=15) : ";
	cin>>n;
	if(n<1 || n>15)
	{
		cout<<"Wrong input, No. of jobs should be less than 15..\n Exitting";
		getch();
		exit(0);
	}
	cout<<"Enter the processing time and their respective probabilities ";
	for(int i=1;i<=n;i++)
	{
		cout<<"\nEnter the processing time and its probability of "<<i<<" job for machine A : ";
		cin>>a[i]>>p[i];
		cout<<"\nEnter the processing time and its probability of "<<i<<" job for machine B : ";
		cin>>b[i]>>q[i];

		//Calculate the expected processing times of the jobs for the machines:
		a1[i] = a[i]*p[i];
		b1[i] = b[i]*q[i];
	}
	cout<<"\nEnter the two job blocks (two numbers from 1 to "<<n<<") : ";
	cin>>group[0]>>group[1];

	cout<<"\nEnter the break down intervals : ";
	cin>>bd1>>bd2;
	cout<<"\nEnter the Rental cost of machine A : ";
	cin>>cost_a;
	cout<<"\nEnter the Rental cost of machine B : ";
	cin>>cost_b;
	//Function for expected processing times for two job blocks
```

```
	time_for_job_blocks();
	int t = n-1;
	schedule(t,1);
	//Calculating In-Out times
	inout_times(job_schedule_final);
	//Calculating revised processing times for both the machines
	//That is updating a1[], and b1[]
	update();

	//REpeat the process for all possible sequences
	for(int k=1;k<=n;k++)   //Loop of all possible sequences
	{
		for(int i=1;i<=n;i++)
		{
			optimal_schedule_temp[i]=job_schedule_final[i];
		}
		int temp = job_schedule_final[k];
		optimal_schedule_temp[1]=temp;
		for(i=k;i>1;i--)
		{
			optimal_schedule_temp[i]=job_schedule_final[i-1];
		}

		//Calling inout_times()
		int flag=0;
		for(i=1;i<n;i++)
		{
			if(optimal_schedule_temp[i]==group[0]                    &&
optimal_schedule_temp[i+1]==group[1])
			{
				flag=1;
				break;
			}
		}
		if(flag==1)
		{
			inout_times(optimal_schedule_temp);
			ta[k]=a1_out[n]-a1_in[1];
			tb[k]=b1_out[n]-b1_in[1];
```

ISSN 2222-1905 (Paper) ISSN 2222-2839 (Online)

```
                        if(tb[k]<tb[k-1])
                        {
                                //copy optimal_schedule_temp to optimal_schedule
                                for(int j=1;j<=n;j++)
                                {
                                        optimal_schedule[j]=optimal_schedule_temp[j];
                                }
                        }
                }
        }
        float smalla = ta[1];
        float smallb = tb[1];
        for(int ii=2;ii<=n;ii++)
        {
                if(smalla>ta[ii])
                        smalla = ta[ii];
                if(smallb>tb[ii])
                        smallb = tb[ii];
        }
        clrscr();
        cout<<"\n\n\n\n\n\n\n\n\t\t\t    #####THE SOLUTION##### ";
        cout<<"\n\n\t****************************************************************";
        cout<<"\n\n\n\t    Optimal Sequence is : ";
        for (ii=1;ii<=n;ii++)
        {
                cout<<optimal_schedule[ii]<<"   ";
        }
        cout<<"\n\n\t    The smallest possible time span for machine A is : "<<smalla;
        cout<<"\n\n\t    The smallest possible time span for machine B is : "<<smallb;
        cost = cost_a*smalla+cost_b*smallb;
        cout<<"\n\n\t    Total Minimum Rental cost for both the machines is : "<<cost;
        cout<<"\n\n\n\t****************************************************************";
        getch();
}

void time_for_job_blocks()
{
        //The expected processing times for two job blocks are
        if(b1[group[0]]<a1[group[1]])
```

ISSN 2222-1905 (Paper) ISSN 2222-2839 (Online)

```
        {
                min = b1[group[0]];
        }
        else
        {
                min = a1[group[1]];
        }
        a1_jb = a1[group[0]]+a1[group[1]] - min; //(b1[k]<a1[m])?b1[k]:a1[m];
        b1_jb = b1[group[0]]+b1[group[1]] - min; //(b1[k]<a1[m])?b1[k]:a1[m];
        getch();
}

void update()
{
        for(int i=1;i<=n;i++)
        {
                if(a1_in[i]<=bd1 && a1_out[i]<=bd1 || a1_in[i]>=bd2 && a1_out[i]>=bd2)
                {
                        a1_t[i] =a1_t[i];
                }
                else
                {
                        a1_t[i] += (bd2-bd1);
                }
                if(b1_in[i]<=bd1 &&b1_out[i]<=bd1 || b1_in[i]>=bd2 && b1_out[i]>=bd2)
                {
                        b1_t[i] =b1_t[i];
                }
                else
                {
                        b1_t[i] += (bd2-bd1);
                }
        }
        //Putting values of a1_t and b1_t into a1 and b1 with proper order of jobs
        for(i=1;i<=n;i++)
        {
                a1[job_schedule_final[i]] = a1_t[i];
                b1[job_schedule_final[i]] = b1_t[i];
        }
```

```
        time_for_job_blocks();

        int t = n-1;
        schedule(t,1);
}
void inout_times(int schedule[])
{
        for(int i=1;i<=n;i++)
        {
                //Reorder the values of a1[] and b1[] according to sequence
                a1_t[i] = a1[schedule[i]];
                b1_t[i] = b1[schedule[i]];
        }
        for(i=1;i<=n;i++)
        {
                if(i==1)
                {
                        a1_in[i]=0.0;
                        a1_out[i] = a1_in[i]+a1_t[i];
                        b1_in[i] = a1_out[i];
                        b1_out[i] = b1_in[i]+b1_t[i];
                }
                else
                {
                        a1_in[i]=a1_out[i-1];
                        a1_out[i] = a1_in[i]+a1_t[i];
                        if(b1_out[i-1]>a1_out[i])
                        {
                                b1_in[i] = b1_out[i-1];
                                b1_out[i] = b1_in[i]+b1_t[i];
                        }
                        else
                        {
                                b1_in[i] = a1_out[i];
                                b1_out[i] = b1_in[i]+b1_t[i];
                        }
                }
        }
}
```

```
int js1=1,js2=n-1;
void schedule(int t, int tt)
{
	if(t==n-1)
	{
		js1=1; js2=n-1;
	}
	if(t>0 && tt==1)
	{
		for(int i=1,j=1;i<=n;i++,j++) //loop from 1 to n-1 as there is one group
		{
			if(i!=group[0]&&i!=group[1])
			{
				a1_temp[j] = a1[i];
				b1_temp[j] = b1[i];
				job_temp[j] = i;
			}
			else if(group[0]<group[1] && i==group[0])
			{
				a1_temp[j] = a1_jb;
				b1_temp[j] = b1_jb;
				job_temp[j] = -1;
			}
			else
			{
				j--;
			}
		}
		//Finding smallest in a1
		float min1= 32767;
		int pos_a1;
		for(j=1;j<n;j++)
		{
			if(min1>a1_temp[j])
			{
				pos_a1 = j;
				min1 = a1_temp[j];
			}
		}
```

ISSN 2222-1905 (Paper) ISSN 2222-2839 (Online)

```
        //Finding smallest in b1
        float min2= 32767;
        int pos_b1;
        for(int k=1;k<n;k++)
        {
                if(min2>b1_temp[k])
                {
                        pos_b1 = k;
                        min2 = b1_temp[k];
                }
        }
        if(min1<min2)
        {
                job_schedule[js1] = job_temp[pos_a1];
                js1++;
                a1_temp[pos_a1]=32767;
                b1_temp[pos_a1]=32767;
        }
        else
        {
                job_schedule[js2] = job_temp[pos_b1];
                js2--;
                a1_temp[pos_b1]=32767;
                b1_temp[pos_b1]=32767;
        }
}
else if(t>0 && tt!=1)
{
        //Finding smallest in a1
        float min1= 32767;
        int pos_a1;
        for(int i=1;i<n;i++)
        {
                if(min1>a1_temp[i])
                {
                        pos_a1 = i;
                        min1 = a1_temp[i];
                }
        }
```

```
        //Finding smallest in b1
        float min2= 32767;
        int pos_b1;
        for(i=1;i<n;i++)
        {
                if(min2>b1_temp[i])
                {
                        pos_b1 = i;
                        min2 = b1_temp[i];
                }
        }
        if(min1<min2)
        {
                job_schedule[js1] = job_temp[pos_a1];
                js1++;
                a1_temp[pos_a1]=32767;
                b1_temp[pos_a1]=32767;
        }
        else
        {
                job_schedule[js2] = job_temp[pos_b1];
                js2--;
                a1_temp[pos_b1]=32767;
                b1_temp[pos_b1]=32767;
        }
}
t--;
if(t!=0)
{
        schedule(t, 2);
}
//final job schedule
int i=1;
while(job_schedule[i]!=-1)
{
        job_schedule_final[i]=job_schedule[i];
        i++;
}
job_schedule_final[i]=group[0];
```

```
i++;
job_schedule_final[i]=group[1];
i++;
while(i<=n)
{
        job_schedule_final[i]=job_schedule[i-1];
        i++;
}
```

11. Numerical Illustration

Consider 5 jobs and 2 machines problem to minimize the rental cost. The processing times with their respective associated probabilities are given as follows. Obtain the optimal sequence of jobs and minimum rental cost of the complete set up, given rental costs per unit time for machines M_1 & M_2 are 16 and 14 units respectively, and jobs (2, 5) are to be processed as an equivalent group job with the break-down interval as (5,10).

Jobs	Machine M_1		Machine M_2	
i	A_i	p_i	B_i	q_i
1	11	0.1	8	0.2
2	15	0.3	11	0.2
3	14	0.1	15	0.1
4	17	0.2	16	0.2
5	12	0.3	18	0.3

Solution

Step 1**:** The expected processing times A_i' and B_i' on machine A and B are as in table 1.

Step 2: The processing times of equivalent job block β = (2,5) by using *Maggu* and *Das* criteria are **(show in table 2)** given by

$A_\beta' = 4.5 + 3.6 - 2.2 = 5.9$

and $B_\beta' = 2.2 + 5.4 - 2.2 = 5.4$

Step 3**:** Using *Johnson's* two machines algorithm, the optimal sequence is

S = 1, 3, β, 4 .i.e. S = 1, 3, 2, 5, 4

Step 4: The in-out flow table for the sequence S = 1- 3- 2- 5- 4 is prepared **(Shown in table 3)**.

Step 5: On considering the effect of break-down interval (5, 10), the revised processing times A_i'' and B_i'' of machines A and B are calculated **(Shown in table 4)**.

Step 6**:** The new processing times of equivalent job block β = (2,5) by using *Maggu* and *Das* criteria are **(Shown in table 5)** given by

$A_\beta'' = 9.5 + 8.6 - 7.2 = 10.9$ and $B_\beta'' = 7.2 + 5.4 - 7.2 = 5.4$

Step 7: Using *Johnson's* two machines algorithm, the optimal sequence is S_1= 1, 3, β, 4 .i.e.

$S_1 = 1 - 3 - 2 - 5 - 4$

Step 8: The processing time of *1st* job on S_1 = 1.1, .i.e. α = 1.1.

Step 9: The other optimal sequences for minimizing rental cost are

$S_2 = 2 - 1 - 3 - 5 - 4$, $S_3 = 3 - 1 - 2 - 5 - 4$, $S_4 = 4 - 1 - 3 - 2 - 5$, $S_5 = 5 - 1 - 3 - 2 - 4$

Step 10: The in-out flow tables for sequences S_1, S_3 and S_4 having job block (2, 5) are as shown in **table 6, 7 and 8**.

For S_1= 1 – 3 – 2 – 5 – 4

Total time elapsed on machine A = $t_1(S_1)$ = 24.0

Total time elapsed on machine B = $t_2(S_1)$ = 29.2

Utilization time of 2nd machine (B)= U_1 = 29.2 – 1.1 = 28.1.

For S_3 = 3 – 1 – 2 – 5 – 4

Total time elapsed on machine A = $t_1(S_3)$ = 24.0

Total time elapsed on machine B = $t_2(S_3)$ = 29.2

Utilization time of 2nd machine (B)= U_2 = 29.2 – 1.4 = 27.8.

For S_4 = 4 – 1 – 3 – 2 – 5

Total time elapsed on machine A = $t_1(S_4)$ = 24.0

Total time elapsed on machine B = $t_2(S_4)$ = 29.4

Utilization time of 2nd machine (B)= U_3 = 29.4 – 3.4 = 26.0

The total utilization of machine A is fixed 24.0 units and minimum utilization of B is 26.0 units for the sequence S_4. Therefore the optimal sequence is **S_4 = 4 – 1 – 3 – 2 – 5**.

Therefore minimum rental cost is = 24.0 x 16 + 26.0 x 14 = 748 units.

12. Remarks

1. In case the break-down interval criteria is not taken in consideration then result tally with Singh T.P. and Gupta Deepak [12].

2. The study may further be extended if parameters like set up time, transportation time etc. are taken into consideration.

References

Johnson, S.M.(1954), Optimal two & three stage production schedules with set up times includes, Nav. Res. Log. Quart. Vol 1. pp 61-68.

W.E.Smith (1956), Various optimizers for single stage production, Naval Research logistic 3, pp 89-66.

Ignall E and Scharge, L(1965), Application of branch and bound technique to some flow shop scheduling problems, Operation Research 13, 400-412.

Bagga P.C. (1969), Sequencing in a rental situation, Journal of Canadian Operation Research Society.7 152-153.

Maggu, P.L & Dass. G (1977), Equivalent jobs for job block in job sequencing, Opsearch, Vol. 14 No. 4, pp. 277-281.

Szware, W. (1977), Special cases of flow shop problem, Naval Research Logistics Quarterly, 24, pp 483-492.

S. Szwarc (1983), The flow shop problem with mean completion time criterion, AIIE Trans. 15, pp 172-176.

Singh , T.P. (1985), 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).

Adiri; I., Bruno, J., Frostig, E. and Kan; R.A.H.G(1989), Single machine flow time scheduling with a single break-down, Acta Information, Vol.26(No.7) : 679-696.

Akturk, M.S. and Gorgulu, E (1999), Match up scheduling under a machine break-down, European journal of operational research: 81-99.

Chander Shekharn, K, Rajendra, Deepak Chanderi(1992), An efficient heuristic approach to the scheduling of jobs in a flow shop, European Journal of Operation Research 61, 318-325.

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

Chandramouli, A.B.(2005), Heuristic approach for N job 3 machine flow shop scheduling problem involving transportation time, break-down time and weights of jobs, Mathematical and Computational Application, Vol.10 (No.2):301-305.

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

Belwal, O.K. and Mittal Amit(2008), N jobs machine flow shop scheduling problem with break-down of machines, transportation time and equivalent job block, Bulletin of Pure & Applied Sciences-Mathematics, Jan-June,2008 Source Volume 27 Source issue: 1.

Notes

Note 1. Break-down time interval (a, b) for which the machines remain unavailable is deterministic in nature. The break-down interval length L = b - a is known**.**

Note 2. Idle time of *1st* machine is always zero i.e. $\sum_{i=1}^{n} I_{i1} = 0.$

Note 3. Idle time of *1st* job on *2 nd* machine $\left(I_{i2}\right)$ = Expected processing time of 1st job on machine = $A_i^{'}$.

Note 4. Rental cost of machines will be minimum if idle time of *2nd* machine is minimum.

Table 1. The expected processing times $A_i^{'}$ and $B_i^{'}$ on machine A and B are

Jobs	$A_i^{'}$	$B_i^{'}$
1	1.1	1.6
2	4.5	2.2
3	1.4	1.5
4	3.4	3.2
5	3.6	5.4

Table 2. The processing times after applying equivalent job block β = (2, 5) are

Jobs	$A_i^{'}$	$B_i^{'}$
1	1.1	1.6
β	5.9	5.4
3	1.4	1.5
4	3.4	3.2

Table 3. The in-out flow table for the sequence S = 1- 3- 2- 5- 4 is

Jobs	A	B
	In-Out	In-Out

1	0.0- 1.1	1.1 – 2.7
3	1.1 – 2.5	2.7 – 4.2
2	2.5 – 6.9	6.9 – 9.1
5	6.9 – 10.5	10.5 – 15.9
4	10.5 – 13.9	15.9 – 19.1

Table 4. The revised processing times A_i'' and B_i'' of machines A and B are

Jobs	A_i''	B_i''
1	1.1	1.6
2	9.5	7.2
3	1.4	1.5
4	3.4	3.2
5	8.6	5.4

Table 5. The new processing times of equivalent job block β = (2,5) after break-down effect are

Jobs	A_i''	B_i''
1	1.1	1.6
β	10.9	5.4
3	1.4	1.5
4	3.4	3.2

Table 6. The in-out flow tables for sequence S_1= 1 – 3 – 2 – 5 – 4

Jobs	A	B
	In-Out	In-Out
1	0.0- 1.1	1.1 – 2.7
3	1.1 – 2.5	2.7 – 4.2
2	2.5 – 12.0	12.0 – 19.2
5	12.0 – 20.6	20.6 – 26.0
4	20.6 – 24.0	26.0 – 29.2

Table 7. The in-out flow tables for sequence S_3 = 3 – 1 – 2 – 5 – 4

Jobs	A	B
	In-Out	In-Out
3	0.0- 1.4	1.4 – 2.9
1	1.4 – 2.5	2.9 – 4.5
2	2.5 – 12.0	12.0 – 19.2
5	12.0 – 20.6	20.6 – 26.0

4	20.6 – 24.0	26.0 – 29.2

Table 8. The in-out flow tables for sequence $S_4 = 4 - 1 - 3 - 2 - 5$

Jobs	A	B
	In-Out	In-Out
4	0.0- 3.4	3.4 – 6.6
1	3.4 – 4.5	6.6 – 8.2
3	4.5 – 5.9	8.2 – 9.7
2	5.9 – 15.4	15.4 – 22.6
5	15.4 – 24.0	24.0 – 29.4