Optimization of Realistic Multi-Stage Hybrid Flow Shop Scheduling Problems with Missing Operations Using Meta-Heuristics

M. Saravanan[*1], S. Sridhar[#2], N. Harikannan[#3]
* Principal,
C.R Engineering College, Alagar Kovil, Madurai District, Tamil Nadu, India,
[1]drmsaravanan@yahoo.com
[#]Department of Mechanical Engineering,
PSNA College of Engineering & Technology, Dindigul District, Tamil Nadu, India.
[2]sri_2855@yahoo.co.in, [3]kannan_hari_n@yahoo.com

***Abstract*— A Hybrid flow shop scheduling is characterized 'n' jobs 'm' machines with 'M' stages by unidirectional flow of work with a variety of jobs being processed sequentially in a single-pass manner. The paper addresses the multi-stage hybrid flow shop scheduling problems with missing operations. It occurs in many practical situations such as stainless steel manufacturing company. The essential complexity of the problem necessitates the application of meta-heuristics to solve hybrid flow shop scheduling. The proposed Simulated Annealing algorithm (SA) compared with Particle Swarm Optimization (PSO) with the objective of minimization of makespan. It is show that the SA algorithm is efficient in finding out good quality solutions for the hybrid flow shop problems with missing operations.**

Keywords - Hybrid flow shop; Makespan; Meta-heuristics; Missing Operations.

I. INTRODUCTION

The hybrid flow shop is a generalized flow shop with parallel machine environment of 'n' jobs 'm' machines with 'M' stages. Each stage encompasses a number of parallel identical machines at each stage. The layout of hybrid flow shop environment is shown in fig. 1.The entire job has to be processed first at stage 1, then stage 2 and so on. The scheduling problem in a multi-stage hybrid flow shop has been the subject of considerable research. All the literatures on this focus that each job has to be performed on all the stages, i.e., there are no missing operations for a job at any stage. The missing operations usually exist in many real-life production systems. Chao-Tang Tseng et al. [1] proposed the first paper of two stage hybrid flow shop with missing operations. The first stage contains only one machine while the second stage consists of two identical machines. Some jobs have to be processed on both stages while others need to be only processed on the second stage. Similarly a Four Drawer Furniture Component (4DFC) was investigated by Sridhar et al. [2] in a stainless steel furniture manufacturing company. The system analysed is composed of five workstations (stages) used for punching, bending, and spot welding, power pressing, and drilling in series. The number of machines in punching, bending, spot welding, power pressing and drilling stages are 5, 8, 3, 5 and 1 respectively. All jobs are performed by one (or) two (or) three machines depending upon the necessity. The missing operation usually occurs in all the jobs. This is identified as a hybrid flow shop environment with missing operations. As per the direction of Chao-Tang Tseng et al. [1] in this paper extent to multi-stage hybrid flow shop scheduling problem with 0%, 20% and 40% missing operations with the objective function of minimization of makespan.

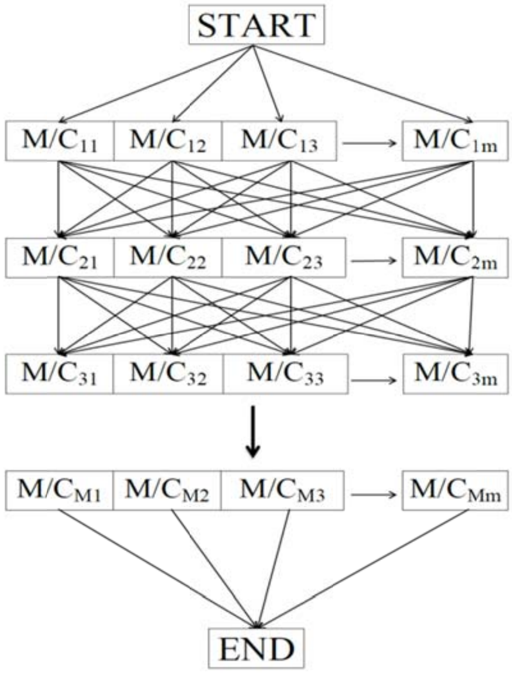

Fig. 1. Layout of Hybrid Flow Shop

II. LITERATURE REVIEW

The number of real world scenario problems is available in hybrid flow shop. It is confined to both in manufacturing and service industries [3]. Many of the researchers have only concentrated both on two stage and multi-stage hybrid flow shop scheduling with single objective of minimizing the makespan. Very few researchers have addressed the assumption of complex hybrid flow shop scheduling problem with missing operations, break down, idleness, earliness, setup times and tardiness.

ToshifumiUetake et al. [4] addressed a two stage hybrid flow shop scheduling problem in which the makespan and the maximum work-in-process are formulated as criteria for manufacturing performance. Lin Wang et al. [5] investigated a difficult scheduling problem on a specialized two-stage hybrid flow shop with multiple processors that appears in semiconductor manufacturing industry. The first and second stages process serial jobs and parallel batches. The objective considered that the minimization of makespan for the job-machine, job-batch, and batch-machine assignments with parallel batch, release time, and machine eligibility constraints. Sridhar et al. [2] compared the real-life company sequence and SA. In which the SA algorithm sequence had given the optimal sequence. Ali Allahverdi and Harun Aydilek [6] addressed m-machine no-wait flow shop scheduling problem with the objective of minimizing total completion time and PA10, PA15, and PA20 algorithms are optimal to the GA algorithm. Ling Wang et al. [7] presented novel decoding method named forward scheduling for the hybrid flow-shop scheduling problem with multiprocessor tasks and objective function of minimize the makespan time. VerenaGondek, [8] addressed the hybrid flow shop scheduling with different additional constraints based on transportation requested with the objective of minimization of total weighted completion time. SinaHakimzadehAbyaneh et al. [9] presented bi-objective hybrid flow shop problem with sequence-dependent setup times and sustainability of limited buffers. EwaFigielska, [10] proposed four heuristic algorithms for two-stage hybrid flow shop with additional renewable resources at each stage and objective function of minimization the makespan time with in small amount of computation time. The 225 papers brief analysis in hybrid flow shop scheduling [3]. Out of all the papers only one paper [1] considered the missing operation in two stage hybrid flow scheduling. The SA algorithm and PSO proposed in 5% and less percentage of papers in present. Ruben Ruiz et al. [11] proposed the m-machine no-wait flow shop problem with the objective of minimizing the maximum lateness. The dominance rule shows great potential for instances the processing and setup times were tightly distributed. Ali Allahverdi and Fawaz, [12] addressed the bi-criteria of two-stage assembly flow shop scheduling problem with a weighted sum of makespan and mean completion time. The experiments reveal that SA performed better than SDE and SA consumes less CPU time than both SDE and Ant Colony Optimization. The hybrid immune algorithm based on the features of artificial immune systems and iterated greedy algorithms proposed in hybrid flow shop scheduling problems to minimize the makespan time [13]. The experiment results clearly reveal that the hybrid immune algorithm is highly effective and efficient as compared to three state-of-the-art meta-heuristics on the same benchmark instances. WalidBesbes et al. [14] addressed in hybrid flow shop scheduling with machines which are not continuously available due to preventive maintenance tasks. Satheesh Kumar et al. [15] proposed PSO algorithm for clearance between the machines in the design of loop layout. The PSO was compared with the differential evolution algorithm and genetic algorithm for bench mark problems. The clearance between the machines was also considered in the design of loop layout. The PSO optimization gave optimal result to selecting the best layout. Hongcheng Liu et al. [16] applied hybrid PSO with Estimation of Distribution Algorithm (EDA) to solve permutation flow shop scheduling problem to Minimization-of-Waiting-time Local search (MWL). The computational experiment on different benchmark problems in permutation flow shop, in which two new best known solutions have been found and superiority of PSO-EDA in terms of accuracy. GhasemMoslehi and MehdiMahnam, [17] approached a hybridization of the PSO and local search algorithm to solve the multi-objective in flexible job-shop scheduling problem. The results displayed that the hybrid PSO algorithm satisfactorily captures the multi-objective flexible job-shop problem and competed well with similar approaches. Chakravarthy and Rajendran, [18] proposed the development of a heuristic based on the SA algorithm for scheduling in a flow shop with the objective of minimizing the makespan and maximum tardiness of a job. The computational evaluation indicated that the heuristic based SA performed better than the heuristic. Mousavi et al. [19] presented SA algorithm for hybrid flow shop scheduling problem to minimize the makespan and total tardiness. The results obtained that the SA algorithm is more effective and efficient. Jolai et al. [20] proposed FSA (fuzzy simulated annealing) for no-wait two-stage flexible flow shop scheduling to minimize the makespan and maximum tardiness of job. Mirsanei et al. [21] developed NSA (noval simulated algorithm) for sequence dependent setup time based hybrid flow shop scheduling problem to minimize the makespan. The results show that NSA outperforms both random key genetic algorithm and immune algorithm. Hui-Mei Wang et al. [22] presented SA algorithm for hybrid flow shop scheduling problem to minimize the makespan time. The SA algorithm was an efficient approach in solving hybrid flow shop scheduling problem with multiprocessor tasks for complex problems. AshwaniDhingra and PankajChandna [23] addressed hybrid SA algorithm for multi objective flow shop scheduling problem. A heuristic-based hybrid simulated annealing (HSA) reveals that obtained the near optimal solutions within a reasonable time. ShuaiTianping et al. [24] considered SA algorithm for hybrid flow shop scheduling problem to minimize the number of tardy jobs. The numerical results indicated the SA was feasible and efficient.

III. Objective Function

The paper addresses a hybrid flow shop problem with missing operations for ‘n’ jobs with ‘m’ machines and ‘M’ stages with an objective of minimizing the makespan time. The hybrid flow shop is characterized by unidirectional flow of work with a variety of jobs being processed sequentially in a one-pass manner. The processing times of all the jobs are well known in advance and all the jobs are processed in the same order at various machines. It is difficult to suggest a sequence that will optimize the makespan time. So the task involves the use of appropriate algorithm to optimize the sequence so as to achieve minimum makespan time. The problem can be defined as minimizing the makespan as below,

$$\boldsymbol{Minimization\ of\ C_{max} = Minimization\ of\ Makespan = Max\left\{C_j, j = 1, 2 \ldots \ldots \ldots n\right\}}$$

A. Assumptions

The problem has the following assumptions:

* All jobs are available at zero time.
* Machines are always available without breakdown.
* The processing time of each job on each machine is known in advance.
* Setup times and withdrawal times are negligible.
* Material Handling duration and charges are negligible.
* Pre-emption is not allowed.

B. Notations

C_{max}	:	Makespan time in seconds
J	:	job identifier (j=1 to n)
M	:	Numbers of stages
m	:	Numbers of parallel identical machines on each stages
n	:	Numbers of jobs
q	:	Percentage of missing operations

IV. PROBLEM ENVIRONMENT

This paper investigated the manufacturing sequence for three drawer vertical filing cabinets in the leading stainless steel furniture manufacturing company, Chennai, India. The system analysed is composed of five workstations (stages) consisting of three punching machines, five bending machines, two spot welding machines, two power press and one drilling machine in series. The three drawer vertical filing cabinet consists of 20 subcomponents. Each subcomponent considers as a job. Each job has fixed processing time on stages as given in the table I. All the 20 jobs are performed by either one (or) more machines. The missing operations occur in all the jobs. All the jobs have to follow the same sequence 1, 2 ... M. The company production sequence is done manually, while performed as j = 1, 2, 3 …….. n for three drawer vertical filing cabinet. This problem environment is termed multi-stage hybrid flow shop scheduling problem with missing operations.

Table I

PROCESSING TIMES OF JOBS (SECONDS) AT EACH STAGES

JOBS / STAGES	j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20
M1	50	78	20	32	54	54	70	60	32	20	0	0	45	0	0	15	0	0	0	0
M2	44	66	30	0	80	80	0	0	77	23	0	0	91	0	0	35	0	0	0	0
M3	0	80	64	0	0	92	0	0	34	34	0	0	0	0	0	0	0	0	0	0
M4	0	0	0	0	0	0	0	58	0	0	7	28	0	14	78	0	9	8	12	10
M5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	42	0	0	0	0

V. SIMULATED ANNEALING ALGORITHM

The SA algorithm is a powerful optimization technique proposed by Kirkpatrick et al (1983). This technique attracts much attention, because of its ability to find the global optimum. It can escape from local optimum solutions of difficult combinatorial optimization problems without specific structure. It can easily deal with nonlinear objective functions and complex constraints. The concept of SA allows the direct inclusion of such constraints, generally due to the fact that the SA algorithm examines one trial solution at a time. Accordingly the complex constraints can be externally added in the core of the algorithm without affecting the main optimization procedure. These features constitute the main advantages of the SA algorithm in comparison with the traditional optimization methods. The SA algorithm simulates the procedure of gradually cooling a metal, until the energy of the system acquires the globally minimal value. Beginning with a high temperature, a metal is slowly cooled, so that the system is in thermal equilibrium at every stage. At each temperature is performed an iterative procedure proposed by Metropolis et al.

This procedure simulates the evolution to thermal equilibrium of a metal for a fixed value of temperature and consists of a sequence of trials. The result of each trial depends only on the result of the previous one (Markov chain). In each trial, the state of an atom is randomly perturbed resulting in a change ΔE to the energy of the system. If $\Delta E<0$, the change is accepted and the new configuration of the system constitutes the starting point for the next trial. If $\Delta E>0$, the change is accepted with a probability given by Boltzmann distribution and temperature corresponds to the current value of temperature. This acceptance rule for new states is referred to as the 'Metropolis criterion'. As the temperature decreases, the Boltzmann distribution concentrates on the lower energy state. Finally, when the temperature approaches to zero, the minimum energy states have a non-zero probability of acceptance.

A. Step by step procedure of SA

This algorithm is a combinatorial optimization techniques based on random evaluation of the objective function has proven to be a good technique in the area of sequencing and scheduling. SA is employed in such a way that it finds best priority sequence through random generation of initial priority sequence set at high temperature and pair-wise exchange perturbation scheme for further improvements. The parameters and steps of the SA algorithm are as follows.

Step 1 *Initialization*
Set AT = 475; FR_CNT = 0; ACCEPT = 0; TOTAL = 0;
Step 2 *Generation of initial solution*
Arbitrarily generate two initial priority job sequences S and B. To find the Makespan time corresponding S and B, and Assign to both M_S and M_B.
Step 3 *Checking termination of SA*
If (FR_CNT = 5) or AT < 20 then go to step 16. Else proceed to step 4.
Step 4 *Generation of neighbours*
Generate number of nearer sequence to S using pair wise perturbation scheme.
Step 5 Find Makespan time of all sequence generated in step 4. Sort the minimum Makespan time and store it in M_S.
Step 6 *Compute ΔS (Ms, Ms')*
If (ΔS<=0) then proceed to step 7. Else go to step 10.
Step 7 Assign S = S', $M_S = M_{S'}$ and ACCEPT = ACCEPT + 1.
Step 8 *Compute ΔB (M_B, M_B')*
If (ΔB<=0) then proceed to step 9. Else go to step 12.
Step 9 Assign B = S', $M_B = M_{B'}$ and FR_CNT = 0, and go to step 12.
Step 10 *Compute P and Sample U*
If U>P then go to step 12. Else proceed to step 11.
Step 11 Assign S = S', $M_S = M_{S'}$ and ACCEPT = ACCEPT + 1.
Step 12 Set TOTAL = TOTAL + 1.
Step 13 If (TOTAL > 4 * n) or (ACCEPT > n), then proceed to step 14. Else go back to step 4.
Step 14 Compute PER = (ACCEPT * 100 / TOTAL).
If PER < 15, then set FR_CNT = FR_CNT + 1.
Step 15 Set AT = AT * 0.9, ACCEPT = 0, TOTAL = 0 and go back to step 3.
Step 16 The algorithm frozen. B contains the best sequence. M_B has the minimum Makespan time.

B. Numerical Illustration of SA for the problem given in Table I

Step 1 *Initialization*
Set Temperature = 475°C, FR_CNT = 0,
Accept = 0, Total = 0.
[Initially setting temperature to 475°C, Freeze count to zero and setting the two counter values Accept and total to zero]
Step 2 *Generation of initial solution*
S and B are two sequences generated randomly
S = 18 9 19 14 2 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16
B = 18 9 19 14 15 20 4 7 13 17 12 1 6 2 5 8 10 3 11 16
Step 3 *Checking termination of SA*
Check If (FR_CNT = 5 (or) Temp = 20°C)
Condition not satisfied, so go to Step 4.
[Checking the termination criteria, whether FR_CNT is equal to 5 or the temperature is equal to 20°C, here termination criteria is not achieved so proceeding step 4]
Step 4 *Generation of neighbours*
Neighbourhoods generated using pair-wise exchange perturbation for seed sequence.
S'= 2 9 19 14 18 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16
[The neighbourhood sequence for **S** is being evaluated by swapping two neighbourhood values in that sequence i.e., S = 18 9 19 14 2 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16 --- swap (18, 2)
S'= **2** 9 19 14 **18** 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16]
Step 5 Evaluating the Makespan Time for S B and S' and stored in M_S, M_B and $M_{S'}$.
$M_S = 21414$
$M_B = 21415$
$M_{S'} = 21414$
Step 6 *Compute ΔS (Ms, Ms')*
Computing ΔS = $M_{S'} - M_S$,

= 21414 - 21414 = 0

Here ΔS ≤ 0 Condition Satisfied. So proceed to Step 7.

[Evaluating the difference between the makespan time of sequence S from makespan time of sequence S' and Checking whether the ΔS value is less than or equal to zero. Here the condition satisfied, so proceed to Step 7]

Step 7 Assign, S = S'= 2 9 19 14 18 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16

$M_S = M_{S'}$ = 21414 and Accept = Accept + 1= 0+1= 1

[Assign the sequence S' to S, store the makespan value of sequence S' in S and increase the counter value accept from 0 to 1]

Step 8 *Compute ΔB (Ms', M_B)*

Computing ΔB = $M_{S'} – M_B$

= 21414 – 21415 = -1 < 0

Here ΔB ≤ 0 Condition Satisfied. So proceed to Step 9.

[Evaluating the difference between the makespan time of sequence B from makespan time of sequence S' and Checking whether the ΔB value is less than or equal to zero. Here the condition satisfied, so proceed to Step 9]

Step 9 Assign, B = S'= 2 9 19 14 18 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16

$M_B = M_{S'}$ = 21415

FR_CNT = 0 and Go to Step 12.

[Assign the sequence S' to B, store the makespan value of sequence S' in B and increase the counter value accept from 0 to 1]

Step 12 Set Total = Total + 1= 0+1= 1.

[Increase the counter value total from 0 to 1]

Step 13 *Check for iteration*

As (total < 4 * n) or (accept < n) proceed to step 4 otherwise, when it satisfies the condition (TOTAL > 4 * n) or (ACCEPT > n), it completes one iteration for set temperature (now T = 475°C) and proceeds to Step 14.

Step 14 Freeze counter increases by one FR_CNT = FR_CNT + 1; on satisfying the conditions stated in 14 and proceed to Step 15.

Step 15 Annealing temperature reduces to 0.9* AT. The TOTAL and ACCEPT are set to zero and then proceed to Step 3.

Step 16 Termination of SA (FR_CNT = 5 (or) Temp = 20°C); B contains the best sequence and M_B contains minimum makespan time.

B = 19 14 2 17 12 3 16 11 7 6 5 15 13 10 9 8 20 1 18 4

M_B = 17668

VI. PSO ALGORITHM

The beginning of the PSO algorithm is done with a population of random solutions. New generations are formed by means of velocity updates. The good solution is searched among the updated generations. The potential solutions are called particles. The particles fly through the multi-dimensional search space and follow the current optimum particles. The PSO is carried out the optimal value for the required number of iterations. All particles in the pool are kept during the whole run. PSO does not combine the survival of the rightist, whereas all other evolutionary algorithms do. Each particle has particular velocity. Particles are carried to new positions with this velocity. The each iteration of fitness value particles are evaluated according to their positions. The communication between the each particle segments its information with others. A particle exchanges its information with the particles in the neighborhood. Therefore, after some number of iterations the swarm loses its diversity and the algorithm converges to the optimal solution.

A. Basic Elements

The basic elements of PSO algorithm is expressed as follows:

a. Particle:

X_i denotes the i^{th} particle in the swarm at iteration t and is represented by n number of dimensions as, $[X_i]^t = [(x_{i1})^t, (x_{i2})^t, \dots \dots \dots (x_{in})^t$, Where $(x_{ij})^t$ is the position value of the i^{th} particle with respect to the j^{th} dimension (j = 1,2,..., n).

b. Population:

pop^t is the set of ρ particles in the swarm at iteration t. $pop^t = [(X_1)^t, (X_2)^t, \dots \dots \dots (X_\rho)^t]$.

c. Permutation:

A New variable $(Лi)^t$ which is a permutation of jobs implied by the particle $(X_1)^t$. It can be described as, $[Л_i] = [(Л_{i1})^t, (Л_{i2})^t, \dots \dots \dots (Л_{in})^t]$, Where $(Л_{ij})^t$ is the assignment of job j of the particle i in the permutation at iteration t.

d. Particle velocity:

$[V_i]^t$ is the velocity of particle i at iteration t. It can be defined as, $[V_i] = [(v_{i1})^t, (v_{i2})^t, \dots \dots \dots (v_{in})^t]$ Where $(v_{ij})^t$ is the velocity of particle i at iteration t with respect to the j^{th} dimension.

e. Personal best:

$(P_i)^t$ represents the best position of the particle i with the best fitness until iteration t, so the best position associated with the best fitness value of the particle i obtained so far is called the personal best. For each particle in the swarm, $(P_i)^t$ can be determined and updated at each iteration t. In a minimization problem with the objective function $f(Л_i)^t$ where $(Л_i)^t$ is the corresponding permutation of particle $(X_i)^t$, the personal best $(P_i)^t$ of the i^{th} particle is obtained such that $f(Л_i)^t \leq f(Л_i)^{t-1}$, where $(Л_i)^t$ is the corresponding permutation of personal best $(P_i)^t$ and $(Л_i)^{t-1}$ is the corresponding permutation of personal best $(P_i)^{t-1}$. To simplify, we denote the fitness function of the personal best as $(f_i)^{pb} = f(Лi1)^t$. For each particle, the personal best is defined as $[P_i] = [(p_{i1})^t, (p_{i2})^t, \dots \dots \dots (p_{in})^t]$, where $(p_{ij})^t$ is the position value of the i^{th} personal best with respect to the j^{th} dimension (j = 1,2,..., n).

f. Global Best:

$(G)^t$ denotes the best position of the globally best particle achieved so far in the whole swarm. For this reason, the global best can be obtained such that $f(Л)^t \leq f(Л_i)^t$ for i = 1,2,...,ρ where $(Л)^t$ is the corresponding permutation of global best $(G)^t$ and $(Л_i)^t$ is the corresponding permutation of personal best $(P_i)^t$. Simplify to denote the fitness function of the global best as $(f)^{gb} = f(Л)^t$. The global best is then defined as $(G)^t = [(g_1)^t, (g_2)^t, \dots \dots \dots (g_n)^t]$, where $(g_j)^t$ is the position value of the global best with respect to the j^{th} dimension (j = 1,2,... n).

g. Termination criterion:

It is a condition that the search process will be terminated. It might be a 750 iteration to terminate the search.

B. Step by step procedure for PSO

Step 1

Initialize a population of n particles generated randomly.

Step 2

Compute fitness value for each particle. The fitness value is better than the best fitness value (p_{ij}^{t-1}). Set current value as the new pbest.

Step 3

Choose particle with the best fitness value of all the particles as the gbest (g_j^{t-1}).

Step 4

For each particle, calculate velocity and position by using the equation,

$$(V_{ij})^t = \left[(v_{ij})^{t-1} + c_1 r_1 \left\{(p_{ij})^{t-1} - (x_{ij})^{t-1}\right\} + c_2 r_2 \left\{(g_j)^{t-1} - (x_{ij})^{t-1}\right\}\right]$$

$$(X_{ij})^t = (x_{ij})^{t-1} + (V_{ij})^t$$

Where,

$(v_{ij})^{t-1}$	= Velocity of particle i at t-1^{th} iteration
$(V_{ij})^t$	= Velocity of particle i at t^{th} iteration
$(x_{ij})^{t-1}$	= Position of particle i at t-1^{th} iteration
$(X_{ij})^t$	= Position of particle i at t^{th} iteration
c_1	= Acceleration factor related to pbest
c_2	= Acceleration factor related to gbest
r_1	= Random number between 0 and 1
r_2	= Random number between 0 and 1
$(g_j)^{t-1}$	= global best position of swarm
$(p_{ij})^{t-1}$	= local best position of particle

Step 5

Update particle velocity and position.

Step 6

Terminate if maximum number of iterations is reached. Otherwise, go to Step 2.

There is a communication between the each particle delivers its information with others. A particle exchanges its information with the particles in the neighborhood. Therefore, after some number of Iterations the swarm loses its diversity and the algorithm converges to the optimal solution. Since PSO consists of simple concepts and mathematical operations with little memory requirements it is fast and appealing in use for many optimization problems. To verify the PSO algorithm, comparisons with simulated annealing algorithm is made. Computational results show that the PSO algorithm is very competitive. Computational results show that the local search can be really guided by PSO.

C. Numerical Illustration of PSO for the problem given in Table I

Initializing 100 particles are generated randomly. Evaluating the fitness function of 100 particles and choose 10 best particles according to their fitness function from 100 particles. Among these 10 particles best particle as gbest according to their fitness function.

Particle 1

Particle 1 = 18 9 19 14 15 20 4 7 13 17 12 1 6 2 5 8 10 3 11 16 = Makespan = 21415

p_{best} = 18 9 19 14 15 20 4 7 13 17 12 1 6 2 5 8 10 3 11 16 = Makespan = 21415

g_{best} = 18 9 19 14 2 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16 = Makespan = 21414

Position to p_{best}

18 9 19 14 15 20 4 7 13 17 12 1 6 2 5 8 10 3 11 16 = Makespan = 21415

Position to g_{best}

18 9 19 14 15 20 4 7 13 17 12 1 6 2 5 8 10 3 11 16

Swap (15, 2) and (15, 17)

18 9 19 14 **2** 20 4 7 13 **15** 12 1 6 **17** 5 8 10 3 11 16

Velocity

$$(V_{ij})^t = [(v_{ij})^{t-1} + c_1 r_1\{(p_{ij})^{t-1} - (x_{ij})^{t-1}\} + c_2 r_2\{(g_j)^{t-1} - (x_{ij})^{t-1}\}]$$

$(V_{ij})^t$ = 0 + [1*0.75 (0)] + [1*0.55 (15,2) (15,17)]

Assume c_1 and c_2 =1; r_1= 0.75 and r_2 = 0.55 generated randomly.

In the first part of the equation 75% probability has to be considered. So, here swapping value is not considered for velocity. In the second part of the equation 55% probability has to be considered. So the minimum of 50% of the changes are taken i.e. only (15, 2) is taken.

$(V_{ij})^t$ = (15,2).

Position

$$(X_{ij})^t = (x_{ij})^{t-1} + (V_{ij})^t$$

$(X_{ij})^t$ = 18 9 19 14 15 20 4 7 13 17 12 1 6 2 5 8 10 3 11 16 + (15, 2)

= 18 9 19 14 **2** 20 4 7 13 17 12 1 6 **15** 5 8 10 3 11 16

Particle 2

Particle 2 = 2 9 19 14 18 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16 = Makespan = 21414

p_{best} = 2 9 19 14 18 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16 = Makespan = 21414

(Compare particle 1 and 2, particle 2 is best particle according to their fitness function. So here Particle 2 is pbest)

g_{best} = 18 9 19 14 2 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16 = Makespan = 21414

Position to p_{best}

2 9 19 14 18 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16 = Makespan = 21414

Position to g_{best}

2 9 19 14 18 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16

swap (18,2)

18 9 19 14 **2** 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16

Velocity

$$(V_{ij})^t = [(v_{ij})^{t-1} + c_1 r_1\{(p_{ij})^{t-1} - (x_{ij})^{t-1}\} + c_2 r_2\{(g_j)^{t-1} - (x_{ij})^{t-1}\}]$$

$(V_{ij})^t$ = (15,2) + [1*0.75 (0)] + [1*0.55 (18,2)]

In the first and second part of the equation 75% and 55% probability has to be considered respectively. So, here swapping value is not considered for velocity.

Position

$$\left(X_{ij}\right)^{t} = \left(x_{ij}\right)^{t-1} + \left(V_{ij}\right)^{t}$$

$(X_{ij})^t$ = 2 9 19 14 18 20 4 7 13 15 12 1 6 17 5 8 10 3 11 16 + (15,2)

= **15** 9 19 14 18 20 4 7 13 **2** 12 1 6 17 5 8 10 3 11 16

The new sequence 1 and 2 are 18 9 19 14 2 20 4 7 13 17 12 1 6 15 5 8 10 3 11 16 and 15 9 19 14 18 20 4 7 13 2 12 1 6 17 5 8 10 3 11 16 respectively for the next iteration. Similarly for all other eight particles, new particles are generated and evaluate the fitness function. This course of action is completed means one iteration is completed. This procedure is carried out for 750 iterations to get the optimum value.

Optimum sequence = 20 11 17 6 19 5 18 3 12 10 2 16 9 14 8 13 7 1 15 4 and their fitness value = 17669.

VII. Computational Results

The proposed SA and PSO algorithms evaluated the performance aspects. The population based both optimization algorithms were coded in java and run on a corei3 processor 2.40 GHz PC with 6 GB memory. The SA and PSO algorithms are tested over randomly selected test set problems. The 5 and 8 stage hybrid flow shop problems were generated with the number of jobs n = 5, 10, 15, 20, 25, 50, 60, 75, 80, 100 and each stage involves of 5 and 8 machines in the experiment. For each problem 100 batch size and 30 instances were generated. The percentages of missing operations q% tested were 0%, 20% and 40%. The processing times are selected as random integers from a uniform distribution p(i,j)_U[1, 99] at all stages for 0% of missing operations and uniform distribution p(i,j)_U[0, 99] at all stages with 20% and 40% of missing operations. Totally, 3600 instances (120 different test sets with each test set 30 instances) observed in the experiment. The Relative Performance Deviation (RPD) computes as the algorithm performance evaluation criterion. The RPD of the solution can be calculated as,

$$\mathbf{RPD} = ((\mathbf{Current\ Algorithm\ Solution} - \mathbf{Best\ Solution}) / \mathbf{Best\ Solution}) * \mathbf{100}$$

The RPD average and maximum values of the solution obtained by the makespan approach and computational time in seconds needed by the algorithms to achieve the best solution. The average RPD values are calculated over 30 instances generated for a specified problem test sets and also observed the maximum RPD values in these 30 instances. The SA and PSO algorithms results of the average value of RPD (Avg) and maximum RPD (Max) are analyzed in Table II to VII for 5 and 8 stages of hybrid flow shop problem test sets with q = 0%, 20% and 40% of missing operations respectively and also indicated the computational time in seconds. It is found that the proposed SA algorithm outperforms PSO in all number of test sets and takes less computation times than PSO in most cases. The computational time increases in SA for large size of problems than PSO.

Table II

Relative Percentage Deviation of SA and PSO in 5 stage test sets (q = 0%)

M	m	n	SA		PSO		Computational Time (Seconds)	
			Avg	Max	Avg	Max	SA	PSO
5	5	5	0	0	1.512	2.80	1	32
		10	0.029	0.283	0.217	0.822	2	37
		15	0.020	0.103	0.688	2.850	3	41
		20	0.087	0.401	0.176	0.819	6	47
		25	0.022	0.057	0.113	0.546	9	52
		50	0.026	0.041	0.312	1.191	33	77
		60	0.033	0.051	0.388	1.565	46	82
		75	0.043	0.100	0.265	1.119	71	97
		80	0.132	0.586	0.346	1.072	81	100
		100	0.020	0.036	0.071	0.348	119	120
		Mean	**0.0412**	**0.1658**	**0.4088**	**1.3132**	-	-
	8	5	0	0	0.647	1.918	1	34
		10	0.444	1.144	0.403	1.099	2	38
		15	0.164	0.394	0.370	1.226	3	43
		20	0.117	0.216	0.382	2.066	6	48
		25	0.107	0.206	0.193	0.634	9	55
		50	0.185	0.508	0.160	0.887	37	81

		60	0.099	0.176	0.308	1.592	51	92
		75	0.123	0.152	0.545	1.808	71	102
		80	0.182	0.420	0.238	1.059	94	107
		100	0.093	0.179	0.399	1.158	127	124
		Mean	**0.1514**	**0.3395**	**0.3645**	**1.3447**	**-**	**-**

Table III

Relative Percentage Deviation of SA and PSO in 8 stage test sets (q = 0%)

M	**m**	**n**	**SA**		**PSO**		**Computational Time (Seconds)**	
			Avg	**Max**	**Avg**	**Max**	**SA**	**PSO**
8	5	5	0	0	0.169	0.348	1	31
		10	0.760	1.115	0.301	1.370	2	36
		15	2.468	4.473	2.447	5.408	3	41
		20	0.248	1.689	0.067	0.219	6	47
		25	0.418	1.126	1.949	3.746	9	55
		50	0.064	0.123	0.089	0.538	38	82
		60	0.006	0.034	0.148	0.418	52	92
		75	0.015	0.057	0.145	0.783	73	102
		80	0.078	0.112	0.347	0.901	93	106
		100	0.036	0.051	0.417	1.272	126	127
		Mean	**0.4093**	**0.878**	**0.6079**	**1.5003**	**-**	**-**
	8	5	0	0	1.038	8.340	1	35
		10	1.761	3.117	0.489	1.550	2	38
		15	1.077	2.266	1.441	5.044	3	44
		20	0.368	1.048	0.883	2.975	6	48
		25	0.623	1.642	1.464	8.173	10	55
		50	0.146	0.425	0.331	1.714	38	82
		60	0.011	0.055	0.297	1.673	51	93
		75	0.026	0.100	0.382	1.185	72	104
		80	0.113	0.169	0.645	2.441	94	108
		100	0.136	0.183	0.199	0.732	128	124
		Mean	**0.4261**	**0.9005**	**0.7169**	**3.3827**	**-**	**-**

Table IV

Relative Percentage Deviation of SA and PSO in 5 stage test sets (q = 20%)

M	**m**	**n**	**SA**		**PSO**		**Computational Time (Seconds)**	
			Avg	**Max**	**Avg**	**Max**	**SA**	**PSO**
5	5	5	0	0	3.895	4.493	1	32
		10	2.220	5.552	5.692	13.606	2	36
		15	0.094	0.306	0.436	2.491	3	41
		20	0.696	1.415	1.544	4.827	6	47
		25	0.940	3.679	1.491	4.790	8	51
		50	0.101	0.170	1.047	2.684	31	74
		60	1.923	2.647	0.843	1.680	43	82
		75	1.641	3.365	2.111	5.426	65	93
		80	0.164	0.949	0.750	2.790	74	97
		100	1.517	2.757	3.270	7.435	114	114
		Mean	**0.9296**	**2.084**	**2.1079**	**5.0222**	**-**	**-**
	8	5	0.134	0.448	2.092	8.861	1	32
		10	2.387	6.385	2.528	8.393	2	37
		15	0.308	0.456	0.279	0.587	4	42
		20	1.343	3.624	3.673	7.220	6	48
		25	1.435	2.502	2.505	7.415	9	52
		50	0.146	0.254	0.277	1.127	33	75
		60	2.493	4.077	1.906	5.236	47	85
		75	1.974	4.633	2.032	6.037	72	99
		80	0.563	3.793	1.410	5.804	83	104
		100	1.293	2.032	3.270	5.889	128	123
		Mean	**1.2076**	**2.8204**	**1.9972**	**5.6569**	**-**	**-**

Table V

Relative Percentage Deviation of SA and PSO in 8 stage test sets (q = 20%)

M	m	n	SA		PSO		Computational Time (Seconds)	
			Avg	Max	Avg	Max	SA	PSO
8	5	5	0	0	3.356	13.382	1	31
		10	0.065	0.314	0.123	0.432	2	36
		15	4.769	7.215	3.472	6.279	3	41
		20	1.151	2.256	1.438	7.738	6	47
		25	1.127	3.462	2.172	5.369	9	56
		50	1.080	2.467	3.025	6.007	38	82
		60	0.042	0.121	0.637	1.372	52	92
		75	0.012	0.033	0.643	1.854	74	102
		80	0.042	0.088	0.385	1.281	94	106
		100	0.154	0.435	0.532	2.219	126	125
		Mean	**0.8442**	**1.6391**	**1.5783**	**4.5933**	-	-
	8	5	0.078	0.784	1.118	3.762	1	35
		10	0.219	1.048	1.236	3.160	2	38
		15	6.223	9.565	6.257	15.720	3	45
		20	1.129	2.332	1.082	4.053	6	48
		25	1.085	1.987	1.146	3.247	10	55
		50	1.119	2.159	4.114	7.685	38	82
		60	0.063	0.158	0.815	4.194	51	94
		75	0.011	0.021	0.606	3.128	72	104
		80	0.161	0.343	0.361	1.779	94	108
		100	0.055	0.146	1.196	3.399	128	126
		Mean	**1.0143**	**1.8543**	**1.7931**	**5.0127**	-	-

Table VI

Relative Percentage Deviation of SA and PSO in 5 stage test sets (q = 40%)

M	m	n	SA		PSO		Computational Time (Seconds)	
			Avg	Max	Avg	Max	SA	PSO
5	5	5	0	0	0.546	3.459	1	31
		10	0.869	2.677	1.549	5.163	2	36
		15	0.013	0.066	0.210	0.395	3	40
		20	0.011	0.029	0.091	0.230	5	45
		25	0.443	1.271	2.250	9.896	8	50
		50	0.059	0.195	0.042	0.155	29	71
		60	0.281	1.160	0.709	3.691	42	79
		75	0.090	0.262	1.332	4.240	64	92
		80	0.138	0.676	0.250	0.955	74	96
		100	2.287	4.204	3.948	10.577	115	113
		Mean	**0.4191**	**1.054**	**1.0927**	**3.8761**	-	-
	8	5	0	0	0.026	0.265	1	32
		10	1.235	2.544	1.231	2.239	2	37
		15	0.188	0.548	0.864	4.385	3	42
		20	0.083	0.230	0.094	0.528	6	48
		25	0.488	0.770	1.841	4.107	10	53
		50	0.160	0.428	0.053	0.210	33	76
		60	0.179	0.562	0.699	3.964	46	86
		75	0.122	0.142	0.245	1.395	71	93
		80	3.020	3.141	3.475	5.386	82	97
		100	3.120	4.365	3.079	8.886	125	120
		Mean	**0.8595**	**1.273**	**1.1607**	**3.1365**	-	-

Table VII

Relative Percentage Deviation of SA and PSO in 8 stage test sets (q = 40%)

M	m	n	SA		PSO		Computational Time (Seconds)	
			Avg	Max	Avg	Max	SA	PSO
8	5	5	0	0	0.235	0.503	1	32
		10	0.265	1.082	0.674	1.205	2	36
		15	3.544	5.568	2.271	5.577	3	42
		20	4.296	8.098	5.937	13.895	6	47
		25	0.138	0.373	0.749	5.701	10	56
		50	0.809	1.837	4.910	11.371	38	82
		60	0.033	0.085	1.632	4.966	52	93
		75	0.046	0.128	1.714	4.826	74	102
		80	0.654	1.575	1.509	3.809	94	106
		100	0.043	0.116	2.268	6.175	126	124
		Mean	**0.9828**	**1.8862**	**2.1899**	**5.8028**	-	-
	8	5	0.317	1.587	0.159	1.587	1	35
		10	0.127	0.450	0.704	3.314	2	38
		15	4.690	6.427	3.152	6.246	3	45
		20	3.013	5.546	5.160	12.122	6	48
		25	0.369	0.510	0.403	1.573	10	55
		50	0.943	1.591	4.433	15.314	38	84
		60	0.038	0.090	0.198	0.401	51	94
		75	0.068	0.253	1.431	2.857	72	104
		80	0.534	1.194	1.778	4.849	94	108
		100	0.161	0.218	2.032	3.960	128	126
		Mean	**1.026**	**1.7866**	**1.945**	**5.2223**	-	-

VIII. CONCLUSION AND FUTURE SCOPE

In this paper, a case study for multi-stage hybrid flow shop with missing operations is carried out for three drawer filing cabin with the objective of minimizing the makespan time. The proposed SA algorithm is compared with PSO, which evaluated using the various sizes and stages of missing operations. It is observed from that the SA algorithm is efficient in finding out good quality solutions at all the stages of missing operations for the hybrid flow shop problems when compared with PSO. The SA computational time is less for less complex problem and it slightly increases with an increase in complexity. Future research efforts need to be focused on the development of hybrid combinatorial algorithms for solving multi-stage hybrid flow shop problems involving setup times, total flow time, inter-stage transport times, release dates and due dates incorporate with missing operations.

REFERENCES

[1] Chao-Tang Tseng, Ching - Jong Liao and Tai-Xiang Liao, "A note on two-stage hybrid flow shop scheduling with missing operations", *Computers & Industrial Engineering*, Vol. 54, No. 3, pp. 695 – 704, 2008.

[2] Sridhar, S., Prabaharan, T., and Saravanan, M., "Optimisation of sequencing and scheduling in hybrid flow shop environment using heuristic approach", *International Journal of Logistics Economics and Globalisation,* Vol. 2, No.4, pp. 331–351, 2010.

[3] Ruben Ruiz and Jose Antonio Vazquez-Rodriguez, "Invited review the hybrid flow shop scheduling problem", *European Journal of Operational Research,* Vol. 205 pp. 1–18, 2010.

[4] ToshifumiUetake, Hitoshi Tsubone and MasaakiOhba, "A production scheduling system in a hybrid flow shop", *International Journal of Production Economics,* Vol. 41, No. 1-3, pp. 395-398, 1995.

[5] I.-Lin Wang, Taho Yang and Yu-Bang Chang, "Scheduling two-stage hybrid flow shops with parallel batch, release time, and machine eligibility constraints", *Journal of Intelligent Manufacturing*, Vol. 23, No. 6, pp. 2271-2280, 2011.

[6] Ali Allahverdi, and HarunAydilek, "Algorithms for no-wait flowshops with total completion time subject to makespan", *International Journal of Advanced Manufacturing Technology,* online, 2013.

[7] Ling Wang, Ye Xu, Gang Zhou, Shengyao Wang and Min Liu, "A novel decoding method for the hybrid flow-shop scheduling problem with multiprocessor tasks", *International Journal of Advanced Manufacturing Technology*, Vol. 59, No. 9-12, pp. 1113-1125, 2011.

[8] VerenaGondek, "Hybrid flow shop scheduling: heuristic solutions and LP-based lower bounds", *Operations Research Proceedings*, Vol. 14, pp. 485-490, 2011.

[9] SinaHakimzadehAbyaneh and M. Zandieh, "Bi-objective hybrid flow shop scheduling with sequence-dependent setup times and limited buffers", *International Journal of Advanced Manufacturing Technology*, Vol. 58, No. 1-4, pp. 309-325, 2011.

[10] EwaFigielska, "Heuristic algorithms for preemptive scheduling in a two-stage hybrid flowshop with additional renewable resources at each stage", *Computers & Industrial Engineering*, Vol. 59, No. 4, pp. 509-519, 2010.

[11] Ruben Ruiz and Ali Allahverdi, "No-wait flow shop with separate setup times to minimize maximum late", *International Journal of Advanced Manufacturing Technology,* Vol. 35, Pp. 551–565, 2007.

[12] Ali Allahverdi and Fawaz S. Al-Anzi, “The two-stage assembly flowshop scheduling problem with bicriteria of makespan and mean completion time”, *International Journal of Advanced Manufacturing Technology,* Vol. 37, pp. 166–177, 2008.
[13] Kuo-Ching Ying,”Minimizing makespan for multistage hybrid flow shop scheduling problems with multiprocessor tasks by a hybrid immune algorithm”, *European Journal of Industrial Engineering,* Vol.6, No.2, pp.199 – 215, 2012.
[14] WalidBesbes, Jacques Teghem and TaicirLoukil, “Scheduling hybrid flow shop problem with non-fixed availability constraints”, *European Journal of Industrial Engineering,* Vol.4, No.4, pp.413– 433, 2010.
[15] Satheesh Kumar. R. M, Asokan. P and Kumanan.S, “Design of loop layout in flexible manufacturing system using non-traditional optimization technique”, *International Journal of Advanced Manufacturing Technology,* Vol. 38, pp. 594–599, 2008.
[16] Hongcheng Liu, Liang Gao and Quanke Pan, “A hybrid particle swarm optimization with estimation of distribution algorithm for solving permutation flow shop scheduling problem”, *Expert Systems with Applications,* Vol. 38, No. 4, pp. 4348–4360, 2010.
[17] GhasemMoslehi and Mehdi Mahnam, “Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search”, *International Journal of Production Economics,* Vol. 129, No. 1, pp. 14–22, 2010.
[18] Karunakaran Chakravarthy and Chandrasekharan Rajendran, “A heuristic for scheduling in a flowshop with the bicriteria of makespan and maximum tardiness minimization”, *Production Planning & Control,* Vol. 10, No. 7, pp.707 –714, 1999.
[19] S. M. Mousavi, M. Zandieh and M. Yazdani, “A simulated annealing/local search to minimize the makespan and total tardiness on a hybrid flow shop”, *International Journal of Advanced Manufacturing Technology,* Vol. 64, No. 1- 4, pp. 369-388, 2013.
[20] F. Jolai, H. Asefi, M. Rabiee and P. Ramezani, “Bi-objective simulated annealing approaches for no-wait two-stage flexible flow shop scheduling problem”, *Scientia Iranica,* Online, 2012.
[21] H. S. Mirsanei, M. Zandieh, M. J. Moayed and M. R. Khabbazi, “A simulated annealing algorithm approach to hybrid flow shop scheduling with sequence-dependent setup times”, *Journal of Intelligent Manufacturing,* Vol. 22, pp. 965–978, 2011.
[22] Hui-Mei Wang, Fuh-Der Chou and Ful-Chiang Wu, “A simulated annealing for hybrid flow shop scheduling with multiprocessor tasks to minimize makespan”, *International Journal of Advanced Manufacturing Technology,* Vol. 53, pp. 761–776, 2011.
[23] AshwaniDhingra and PankajChandna, “Multi-objective flow shop scheduling using hybrid simulated annealing”, *Measuring Business Excellence,* Vol. 14, No. 3, pp.30 – 41, 2010.
[24] ShuaiTianping, YUJinguo and Sun Ling, “A simulated annealing algorithm for the hybrid flow shop scheduling to minimize the number of tardy jobs”, *OR Transactions,* Vol. 17, No. 2, pp. 41-47, 2013.