Hydride Flow Shop to Minimize Makespan and Maximize Tardiness by using Taguchi Optimum Design Process

I Srinivasa Reddy[1] V V Sudheer Babu[2] B Vijay Kumar[3] M Dayakar[4] B Kranthi Kumar[5]
[1,2,3,4,5]Assistant Professor
[1,2,3,4,5]MLRITM College, Dundigal,Hyd-500043

Abstract— The scheduling problem in hybrid flowshops that consist of multi stages in series, each of which has multiple identical and unidentical parallel machines is considered. Here the setup times are sequence dependent and the objective is to minimize makespan and maximum tardiness. To solve such an NP-hard problem, a simulated annealing (SA) algorithm is introduced. The SA (HSA) algorithm is hybridized with a simple local search to the improve the quality of final solution of the SA algorithm. Taguchi method is employed as an optimization technique to extensively tune different parameters and operators of our algorithm. Taguchi orthogonal array analysis is the best parameters for the optimum design process with the least number of experiments, is established a benchmark to draw an analogy between the performance of SA with other algorithms. The objective functions of the minimization of makespan and maximum tardiness are taken into consideration to evaluate the robustness and effectiveness of the HSA. The effects of the increase in the number of jobs on the performance of the algorithm are explored to make sure it is effective in terms of both the acceptability of the solution quality and robustness. The excellency and strength of the HSA are concluded from all the results acquired in various circumstances.

Key words: Taguchi Method, Johnson's Rule, S/N ratio, Orthogonal array L18

I. INTRODUCTION

Scheduling is the allocation of resources over time to accomplish specific tasks. Normally scheduling is done after many other managerial decisions have been made. The scheduling is mainly used in which the finite recourses are available .The characteristic that distinguishes one scheduling system from another in how capacity is considered in determining the scheduling. Scheduling system can use either infinite or finite loading.

A. *Infinite Loading*

Infinite loading occurs when work is assigned to a work center simply based on what is needed over time. No consideration is given directly to whether there is sufficient capacity at the resources required to complete the work, nor is the actual sequence of the work as done by each resource in the work center considered.

B. *Finite Loading*

A finite loading approach actually schedules in detail each resource using the setup and run time required for each order. In essence, the system determines exactly what will be done by each resource at every moment during the working day. If an operation is delayed due to a parts shortage, the order will sit in queue and wait until the part is available from a preceding operation. Theoretically, all schedules are feasible when finite loading is used.

C. *Forward Scheduling*

Another characteristic that distinguishes scheduling system is whether the scheduling is generated forward or backward in time. For this forward-backward dimension, the most common is forward scheduling. Forward scheduling refers to the situation in which the system takes an order and then schedules each operation that must be completed

D. *Backward Scheduling*

Backward scheduling starts from some date in the future (possibly a due date) and schedules required operations in reverse sequence. The backward schedule tells when an order must be started in order to be done by a specific date.

E. *Different Types of Scheduling*

1) Job shop Scheduling
2) Flow shop Scheduling
3) Sequence Dependent
4) Hydride Flow Shop

F. *Hydride Flow Shop*

Hydride flow line to indicate flow lines with the presence of multiple identical machines in parallel at some or all stages, though jobs still require processing at exactly one machine per stage. A flexible flow line is a hybrid or (regular) flow line where at least one job need not be processed on any machines in at least one stage. That is, every job must be processed on at most one machine per stage. A flexible flow line consists of several stages in series. A job may not revisit a stage that it has already visited. Each stage has at least one machine, and at least one stage must have more than one machine. At this point, this structure may be considered a hybrid flow line or a flow line with multiple machines.

G. *Objectivies*

Consider the two independent objectives, namely, the minimization of both maximum completion time (makespan or Cmax) and maximum tardiness (MT). Hybrid flowshops scheduling with makespan objective to HFS-Cmax and with maximum tardiness objective to HFSMT. However, the best sequence with respect to makespan minimization might have a large number of jobs being completed after their due dates. Hence, we also consider the MT minimization. Tardiness of each job is equivalent to the amount of time the job is completed after its due date. The main aim of considering two independent objective functions is to measure the robustness of our proposed HSA algorithm by applying it to two essentially different objective functions and study as to whether the high performance of HSA is transferrable to other objective functions.

II. SIMULATED ANNEALING

Kirkpatrick introduced Simulated Annealing (SA) and Creny considered the analogy between the annealing process of solids and the process of solving combinatorial optimization problems. However, it was originally developed as a simulation model for a physical annealing process of condensed matter (Metropolis et al. 1953).

The table1 below shows how physical annealing can be mapped to simulated annealing.

Thermodynamic Simulation	Combinatorial Optimization
System States	Feasible Solutions
Energy	Cost
Change of State	Neighboring Solutions
Temperature	Control Parameter
Frozen State	Heuristic Solution

Table 1: Relationship between physical annealing and simulated annealing

Using these mappings, any combinatorial optimization problem can be converted into an annealing algorithm.

SA’s major advantage over other methods is an ability to avoid becoming trapped at local minima. The algorithm employs a random search, which not only accepts changes that decrease objective function, *f*, but also some changes that increase it. The latter are accepted with a probability

$$p = \exp(-\frac{\delta f}{T}), \qquad (1)$$

Where δf is the increase in *f* and *T* is a control parameter.

A. *Proposed Simulated Annealing*

Simulated annealing (SA) is a neighborhood search approach designed to obtain a global optimum solution for combinatorial optimization problem. SA starts with an initial solution and iteratively moves towards other existing solutions. In order to reduce the probability of getting trapped in local optima, SA accepts moves to inferior neighboring solution under the control of randomized scheme. More precisely, if a move from current solution s to another inferior neighboring solution s* results in a change ΔC = C(s*) − C(s) in the objective function value, the move is still accepted if R < exp (−ΔC / T), where T is a control parameter, called temperature, and R is a uniform random number between interval (0, 1).

B. *Basic Steps In Algorithm*

- Encoding scheme
- initialization procedure
- Neighborhood search structure (NSS)
- Local search
- Cooling schedule

C. *Neighborhood Search Structure (NSS)*

The main function of neighborhood search structure is to produce a new solution from current solution by making a slight change in it. A variety of NSSs have been applied to scheduling problems. These NSSs must work such that infeasible solutions are avoided. In this work, we look at five different NSSs: Inversion operator

- Single point operator
- Swap operator
- Multi single point operator (MSPO)
- Giant leap

Inversion operator: The positions between two randomly selected cut points are reversed.

Single point operator: The position of one randomly picked job is regenerated.

Swap operator: The positions of two randomly selected jobs are swapped.

Multi Single Point Operator: The introduce an operator which is called multi single point operator (MSPO) by which we intend to create two novels NSSs. MSPO is an extension of single point operator (SPO). In SPO, a randomly selected job in the sequence is relocated into another randomly selected given number of jobs (d) are changed.

Giant Leap: In order to diversify the search space to avoid getting trapped in local optima, we require a specific type of operator able to separate us from current search space and direct us to visit new search space for maintaining the probability of finding a better solution. The procedure of this operator follows the structure of MSPO with this distinctive technical specification that the positions of three randomly selected jobs are relocated into three other randomly selected positions. Semantically, we name this operator “Giant leap” (GL).

D. *Local Search*

In order to reinforce the performance of our algorithm, it is hybridized with a simple form of local search. The procedure of this local search is described as follows: The first job in the sequence x (x1) is relocated into a new random position in the sequence. If this new sequence v results in better objective function, the current solution x is replaced by the new sequence v. This procedure iterates at most for all the subsequent jobs in the sequence x. The local search for current solution x terminates if we observe the first improvement in ith < n. Eventually, the best solution is updated.

E. *Cooling Schedule*

The basic aim of utilizing cooling schedule (i.e., temperature and cooling rate) is to control the behavior of SAs. As previously explained, to avoid local minimum, solutions with worse objective values are probably accepted depending on the temperature. As the procedure proceeds, the temperature is gradually lowered under a certain mechanism called cooling schedule. In general, there are three types of cooling schedule in the literature of SA: linear, exponential, and hyperbolic.

1) Linear cooling rate: Liner $T_i = T_0 - i * \frac{To\_Ti}{N}$; $l = 1, 2, \dots, N$
2) Exponential cooling rate: $T_i = \frac{A.}{i+1} + B$ $A = \frac{(T0\_TF)(N+1)}{N}$,$B = T_0 - A$, $l = 1, 2, \dots, N$
3) Hyperbolic cooling rate: $T_i = 1/2((T_0 - T_f)(1 - tgh(10_i/N - 5)) + T_f$; $l = 1, 2, \dots, N$

reports the corresponding formulas of each standard cooling schedule, where T0, Tf, and N are initial temperature, final (stopping) temperature, and the desired number of temperature between T0 and Tf, respectively (see more details in).In our view, this parameter still needs to be tuned. An appropriate initial temperature should be high enough to create equal opportunity for all states of the search space to be visited, and at the same time, it should not be rather too high to perform quite a lot of unnecessary searches in high temperature. Preliminary tests showed us that the temperature ranging from 10 to 20 is suitable for our problem, and the stopping temperature is fixed at 1.

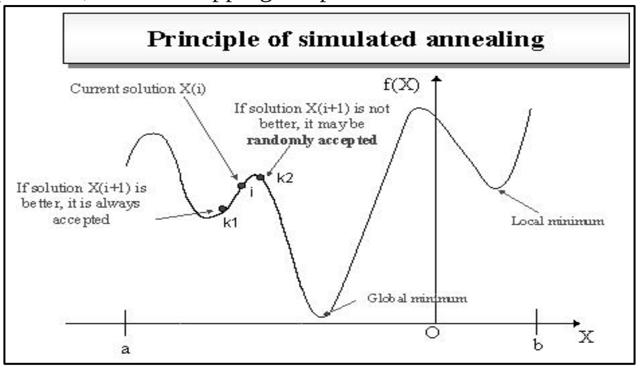
Principle of simulated annealing
Current solution X(i)
f(X)
If solution X(i+1) is not better, it may be randomly accepted
k2
i
If solution X(i+1) is better, it is always accepted
k1
Local minimum
Global minimum
a
O
b
X

Fig. 1: principal of simulated annealing

F. Applications

Some real application of simulated annealing algorithms is listed below:

– Determining the sequence of observations for an automated astronomical telescope
– Computer aided geometric design
– Optimization of econometric statistical functions
– Games with random moves determined by the simulated annealing algorithm
– Arranging connections on chips and switching devices in telephone networks

G. Parameter Tuning

It is known that the great choice of parameters and operators has remarkable influence on the efficiency of simulated annealing. The proper design of SA parameters and operators highly depends on the type of problems. In this section, we are going to study the behavior of the different operators and parameters of the proposed SA. All different combinations of the aforementioned factors and parameters yield many alternative SAs.

H. Introduction to Taguchi Method

Dr. Taguchi of Nippon Telephones and Telegraph Company, Japan has developed a method based on “ORTHOGONAL ARRAY” experiments which gives much reduced “variance” for the experiment with “optimum settings “of control parameters. Thus the marriage of Design of Experiments with optimization of control parameters to obtain BEST results is achieved in the Taguchi Method. "Orthogonal Arrays" (OA) provide a set of well balanced (minimum) experiments and Dr. Taguchi's Signal-to-Noise ratios (S/N), as objective functions for optimization, help in data analysis and prediction of optimum results. Taguchi’s definition of a robust design is: “a product whose performance is minimally sensitive to factors causing variability (at the lowest possible cost)”. Taguchi’s view was that in traditional systems, robustness (or in general, quality) was measured by some performance criteria, such as: meeting the specifications

– % of products scrapped
– Cost of rework
– % defective
– failure rate

These measures of performance are all based on make-and-measure policies. They all come too late in the product development cycle. Robust design is a systematic methodology to design products whose performance is least affected by variations, i.e. noise, in the system

Some statistical tools are necessary to generate robust designs. These are broadly covered in standard courses on Design of experiments. We will study the basic ideas.

Taguchi Method treats optimization problems in two categories,

1) Static Problems
2) Dynamic Problems

1) Static Problems

Generally, a process to be optimized has several control factors which directly decide the target or desired value of the output. The optimization then involves determining the best control factor levels so that the output is at the the target value. Such a problem is called as a "static problem". This is best explained using a P-Diagram which is shown below ("P" stands for Process or Product). Noise is shown to be present in the process but should have no effect on the output! This is the primary aim of the Taguchi experiments - to minimize variations in output even though noise is present in the process. The process is then said to have become robust.

There are 3 Signal-to-Noise ratios of common interest for optimization of Static Problems;

a) Smaller the Better

n = -10 Log_{10} [mean of sum of squares of measured data]

This is usually the chosen S/N ratio for all undesirable characteristics like “defects “etc. for which the ideal value is zero. Also, when an ideal value is finite and its maximum or minimum value is defined (like maximum purity is 100% or maximum Tc is 92K or minimum time for making a telephone connection is 1 sec) then the difference between measured data and ideal value is expected to be as small as possible. The generic form of S/N ratio then becomes,

n = -10 Log_{10} [mean of sum of squares of {measured - ideal}]

b) Larger The better

n = -10 Log_{10} [mean of sum squares of reciprocal of measured data]

This case has been converted to SMALLER-THE-BETTER by taking the reciprocals of measured data and then taking the S/N ratio as in the smaller-the-better case.

c) Nominal the Best

$$n = 10\ \text{Log}_{10}\ \frac{\text{square of mean}}{\text{variance}}$$

This case arises when a specified value is MOST desired, meaning that neither a smaller nor a larger value is desirable. Examples are:

1) Most parts in mechanical fittings have dimensions which are nominal-the-best type.
2) Ratios of chemicals or mixtures are nominally the best type.

e.g. Aqua regains 1:3 of HNO3: HCL Ratio of Sulphur, KNO3 and Carbon in gun powder

3) Thickness should be uniform in deposition /growth /plating /etching.

I. Dynamic Problems

If the product to be optimized has a signal input that directly decides the output, the optimization involves determining the best control factor levels so that the "input signal / output" ratio is closest to the desired relationship. Such a problem is called as a "dynamic problem". This is best explained by a P-Diagram which is shown below. Again, the primary aim of the Taguchi experiments - to minimize variations in output even though noise is present in the process- is achieved by getting improved Linearity in the input-output relationship.

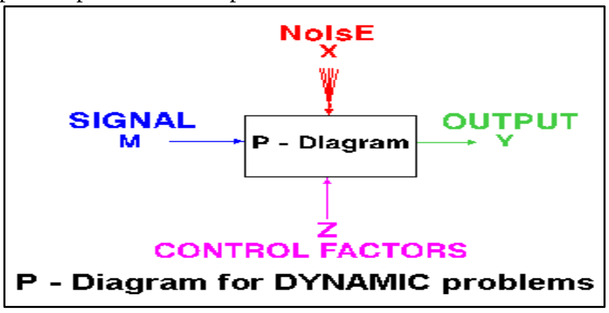

Fig. 2: Dynamic problems

In dynamic problems, we come across many applications where the output is supposed to follow input signal in a predetermined manner. Generally, a linear relationship between "input" "output" is desirable.

For example: Accelerator pedals in cars, volume control in audio amplifiers, document copier (with magnification or reduction) various types of moldings, etc. There are 2 characteristics of common interest in "follow-the-leader" or "Transformations" type of applications, Slope of the I/O characteristics And

(ii) Linearity of the I/O characteristics (minimum deviation from the best-fit straight line) The Signal-to-Noise ratio for these 2 characteristics has been defined as;

1) Sensitivity (slope)

The slope of I/O characteristics should be at the specified value (usually 1). It is often treated as Larger-The-Better when the output is a desirable characteristics (as in the case of Sensors, where the slope indicates the sensitivity).

n = 10 Log10 [square of slope or beta of the I/O characteristics]

On the other hand, when the output is an undesired characteristic, it can be treated as Smaller-the-Better.

n = -10 Log10 [square of slope or beta of the I/O characteristics]

a) Linearity (Larger the Better)

Most dynamic characteristics are required to have direct proportionality between the input and output. These applications are therefore called as "transformations". The straight line relationship between I/O must be truly linear i.e. with as little deviations from the straight line as possible. baki

Variance in this case is the mean of the sum of squares of deviations of measured data points from the best-fit straight line (linear regression).

J. Steps in Taguchi Methodology

Taguchi proposed a standard 8-step procedure for applying his method for optimizing any process,

Step 1 *Confirm quality characteristics of the product or the process in the research.*
Step 2 *Locate factors that affect the certain quality characteristic and their interactions.*
Step 3 *Determine level number of each factor.*
Step 4 *Select the appropriate orthogonal array.*
Step 5 *Assign factors and their interactions to each column of the orthogonal array.*
Step 6 *Conduct experiments based on the orthogonal array distribution.*
Step 7 *Transform experiment results into signal-to-noise ratio (S/N ratio) and determine the optimal level of each factor.*
Step 8 *Conduct confirmation experiment.*

III. PROBLEM FORMULATION

The main aim of problem formulation considering two independent objective functions is to measure the robustness of HSA algorithm by applying it to two essentially different objective functions and study as to whether the high performance of HSA is transferrable to other objective functions. In the first subsection is HFS, with SPT, and (g/2, g/2) Johnson’s rule and in the next subsection, HFS with, EDD and SLACK. The purpose of assessment of effectiveness and competitiveness of different algorithms, SA is compared against some other methods in the literature that have shown the competitive performance in the original papers they have been applied.

A. Objective Functions

As earlier mentioned, we independently consider makespan (Cmax) which belongs to process oriented and maximum tardiness (MT) which belongs to customer oriented to analyze the efficiency of our HSA on each of these objectives against some existing methods.

1) Analysis of Makespan

In this subsection, HSA is compared against SPT, and (g/2, g/2) Johnson’s rule.

Firstly, we analyze the algorithms in terms of selected objective function (makespan) and secondly, in terms of the effects of variable factors, problem size, and numbers of stages, on the performances of them.

2) Analysis of Maximum Tardiness

This section we draw an analogy of our proposed HAS algorithm against EDD, and SLACK to minimize maximum tardiness (MT). To compare the algorithms in the case of HFS-Cmax, RPD measure is employed. In the case of HFS-MT problem, the best solution could be zero (and therefore optimal), so the RPD equation gives a division by zero. Moreover, if the best solution is a small value, the performance measure trivializes an algorithm which obtains a solution slightly worse than the best. Therefore, a different performance ratio is usually used in tardiness cases to avoid these problems which has been named relative deviation index (RDI). RDI is obtained by given formula below:

$$\text{RDI}=\frac{Algsol-Minsol}{Alg_{sol}-Min_{sol}}*100$$

With this measure, an index between 0 and 100 is obtained for each method. The more the RDI is closer to zero, the more the algorithm is preferable. Note that if the worst and the best solutions take the same value, all the methods provide the best (same) solution and, hence, the index value will be 0 (best index value) for all methods. To generate due dates of all n jobs we use an approach, similar to, with the following steps:

Compute total processing time of each job on all g stages.

$P_i=\sum_{t=1}^{g} p_{it} \;\; \forall i \in N$

Where p_{ij} denotes the processing time of job *i* at stage *j*.

Compute average setup times for all possible subsequent jobs and sum it for all g stages.

$S_i=\sum_{t=1}^{g} \frac{\sum_{k=1}^{n} k\neq i}{n-1} \;\; \forall i \in N$

Where s_{ijt} denotes the setup time of job *i* at stage *j* immediately after job *t*.

3. Determine a due date for each job

$d_i = (p_i + s_i) \times (1 + \text{random} \times 3); i \in \{1,2, \dots ,n\}$

Where random is a random number from a uniform distribution over range (0, 1).The results of the experiments, averaged for each combination of n and m results.

B. Data Generation

Data required for a problem consist of the number of jobs (n), the range of processing times (pt), number of stages (g), and whether all stages have the same number of machines or not. Each stage requires data defining how many machines exist at that stage (mt) and the range of the sequence-dependent setup times (SDST), and the ready times. The ready times for stage 1 are set to 0 for all jobs. The ready times of jobs i at stage t are the completion time of job i at stage t-1, so this data need not to be generated.

Number of jobs	20, 50, 80,120
Number of stages	2, 4
Machine distribution	
Constant	2
Variables	U (1, 4)
Processing time	U (1, 99)
SDST	U(1,25) U(1,50) U(1,99)

C. Shortest Processing Time

This is a naïve greedy heuristic that assigns jobs to machines with little or no regard for setup times or the interactions between stages. Because of its simplistic nature, it provides a basis of comparison for the other heuristics. In the SPT Cyclic Heuristic (SPTCH), the jobs are ordered at stage 1 in increasing order of the modified processing times p_i^{-1}. At subsequent stages, jobs are assigned in earliest ready time order. Jobs are assigned to the machine in every stage that allows it to complete at the earliest time as measured in a greedy fashion.

```
1. Create the modified processing times $p_i^{-1}$.
2. Order the jobs in non-decreasing order (SPT) of $p_i^{-1}$.
3. At each stage t=1,…,g, assign job 0 to each machine in that stage.
4. For stage 1:
a. Let bestmc=1.
b. For [i]=1 to n, $i \in S^1$:
For mc=1 to $m^1$:
Place job [i] last on machine mc.
Find the completion time of job [i]. If this time is less on mc than on bestmc,
Let bestmc=mc.
Assign job [i] to the last position on machine bestmc.
5. For each stage t=2,…,g:
a. Update the ready times in stage t to be the completion times in stage t−1.
b. Arrange jobs in increasing order of ready times.
c. Let bestmc=1.
d. For [i]=1 to n, $i \in S^t$:
For mc=1 to $m^t$:
Place job [i] last on machine mc.
Find the completion time of job [i]. If this time is less on mc than on bestmc,
Let bestmc=mc.
Assign job [i] to the last position on machine bestmc.
```

D. Johnson's Rule

1) Scheduling N Jobs on Two Machine

JHONSON'S method has been extended to yield an optimal solution for the n/3case.When flow shop scheduling problems larger than n/3 arise, analytical solutions procedures leading to optimality are not available. The reason for this is that even though the jobs may arrive in static fashion at the first machine, the scheduling problem becomes dynamic and waiting lines start to form in front of machines down streams .at this point it becomes a multistage queuing problem, which is generally solved using simulation techniques.

1) List of operations time for each job on both machines
2) Select the shortest operation time
3) If the shortest time is for the first machine, do the job first, if it is for the second machine do the jobs last. In the case of a tie, do the job on the first machine.
4) Repeat the 2 and 3 for each remaining job until the schedule is complete.

Johnson's Rule finds the optimal makespan solution for $F2||C_{\max}$. Variants have been created, for example by Campbell, for the flow shop with more than two stages. This heuristic is an extension of Johnson's Rule to take into account the setup times. The aggregated first half of the stages and the aggregated last half of the stages are considered to create the order for assignment in stage 1. The value $P_{\sim i}^{-1}$ is the sum of modified processing times for stages 1 to $\lfloor g/2 \rfloor$ and $P_i^{\sim g}$ is the sum over stages $\lfloor g/2 \rfloor+1$ $P_{\sim i}^{-1}$ to $gP_i^{\sim g}$.

```
1) Create the modified processing times and .
2) Let $U = \{j|\bar{p}_j^1 < \bar{p}_j^g\}$ and $V = \{j|\bar{p}_j^1 \geq \bar{p}_j^g\}$.
3) Arrange jobs in U in non-decreasing order of $P_{\sim i}^{-1}$ and arrange jobs in V in non-increasing order of $P_i^{\sim g}$. Append the ordered list V to the end of U.
4) At each stage t=1,…,g, assign job 0 to each machine in that stage.
5) For [i]=1 to n, $i \in S^1$:
    a) For mc=1 to $m^1$:
    b) Place job [i] last on machine mc.
    c) If this placement results in the lowest completion time for job [i], let m=mc.
    d) Place job [i] last on machine m.
6) For each stage t=2,…,g:
    a) Update the ready times in stage t to be the completion times in stage t−1.
    b) Arrange jobs in increasing order of ready times.
    c) For [i]=1 to n, $i \in S^t$:
(1) For mc=1 to $m^t$:
Place job [i] last on machine mc.
If this placement results in the lowest completion time for job [i], let m=mc.
```

(2) Place job [*i*] last on machine *m*.

SA algorithm parameter and operators highly depends on the problem. I have taken parameters and operators valves from previous literature. This approach is not always efficient because it becomes increasingly difficult to carry out investigation when the number of factors becomes significantly high .To reduce the no of experiments using FFEs. In this problem the orthogonal array are used to study a large number of decision variable with small number of experiment

E. Implementation L18

The S/N ratio indicates the amount of variation present in the response variable. The aim is to maximize the signal-to-noise ratio. The associated degree of freedom for these seven factors is 14. So, the selected orthogonal array should have a minimum of 14 rows and seven columns to accommodate the seven factors. From standard table of orthogonal arrays, the L18 is selected as the fittest orthogonal array design which fulfills our all minimum requirements. But this orthogonal array still entails some modifications to adapt itself to the experimental design. Table 1

TRAILS	A	B	C	D	E	F	**G**
1	A1	B1	C1	D1	E1	F1	**G1**
2	A1	B1	C2	D2	E3	F3	G2
3	A1	B2	C1	D3	E3	F2	G3
4	A1	B2	C3	D1	E2	F3	G4
5	A1	B2	C2	D3	E2	F1	G5
6	A1	B2	C3	D2	E1	F2	G5
7	A1	B1	C1	D3	E1	F3	G5
8	A2	B1	C3	D2	E2	F2	G5
9	A2	B2	C2	D1	E3	F2	G1
10	A2	B2	C3	D2	E2	F1	G5
11	A2	B2	C2	D3	E1	F1	G1
12	A2	B2	C3	D2	E3	F3	G2
13	A2	B1	C1	D1	E1	F2	G4
14	A1	B1	C2	D3	E1	F1	G3
15	A1	B1	C3	D2	E2	F3	G5
16	A1	B2	C1	D2	E1	F1	G5
17	A1	B2	C2	D1	E3	F2	G2
18	A1	B2	C2	D1	E2	F3	G1

Table 1: orthogonal array L18

Factor	Symbol	Level	Type
Initial solution	*A*	2	*A*(1) – Random key
			A(2) – Job-based
Encoding Scheme	*B*	2	*B*(1) – John's
			B(2) – SPT
Number of temperatures between T_0 and T_f	*C*	3	*C*(1) – 150
			C(2) – 200
			C(3) – 250
Number of neighborhood search in each temperature	*D*	3	*D*(1) – 30
			D(2) – 60
			D(3) – 100
Initial temperature	*E*	3	*E*(1) – 10
			E(2) – 15
			E(3) – 20
Cooling schedule type	*F*	3	*F*(1) – Linear
			F(2) – Exponential
			F(3) – Hyperbolic
Neighborhood search structure (NSS)	*G*	5	*G*(1) – Swap
			G(2) – SPO
			G(3) – Inversion
			G(4) –MSPO
			G(5) –GL

Table 2: factor sand levels

The problem consists of two 2-level factors, four 3-level factors, and one 5-level factor and L18 is composed of six 3-level factors and one 6-level factor the problem has to be structurally transformed to mold itself to the standard shape of L18. Doing so, for each factor which lacks a certain number of levels in comparison with standard L18, the compensate this lack through the assignment of extra levels in standard L18 to the one of the optionally selected existing levels of the associated factor. For example, consider our factor B which has two levels and lacks one level with regard to standard L18. The extra level in standard L18 is offset by the repetition of the second level in this factor. In the better words, the extra level of standard L18 is assigned to level 2 of factor B to eliminate this gap. Factors A and G should undergo the same procedure to be standardized. Extra levels in standard L18 are assigned to level 1 and level 5 of factors A and G, respectively.

The relative percentage deviation (RPD) as a common performance measure to compare the methods. The best solutions obtained for each instance are calculated by any of the four algorithms. RPD is obtained by given formula below:

$$\text{RPD}=\frac{Alg_{sol}-Min_{sol}}{Min_{sol}}*100$$

Where Algsol is the objective function value obtained for a given algorithms and instance

IV. Results and Discussion

Factors	Signal to noise ratio(S/N)	Relative percentage deviation(RPD)
A1	-30.33	2.896
A2	-37.12	7.425
B1	-31.1	2.02
B2	-35.52	6.12
C1	-33.33	3.01
C2	-34.24	5.608
C3	-37.52	6.22
D1	-34.25	2.98
D2	-35.01	4.962
D3	-36.25	5.534
E1	-33.12	6.765
E2	-30.325	5.99
E3	-29.82	6.25
F1	-35.778	8.92
F2	-33.21	4.92
F2	-32.12	4.02

G1	-36.72	8.25
G2	-29.12	7.02
G3	-41.02	8.12
G4	-42.46	8.88
G5	-46.145	7.25

Table 3: Signal To Noise Ratio And RPD

Factors	Signal to noise ratio(S/N)	Relative deviation index(RDI)
A1	-30.33	3.62
A2	-37.12	8.98
B1	-31.1	4.84
B2	-35.52	10.25
C1	-33.33	5.21
C2	-34.24	8.42
C3	-37.52	9.24
D1	-34.25	3.2
D2	-35.01	5.2
D3	-36.25	6.31
E1	-33.12	8.1
E2	-30.325	8.54
E3	-29.82	7.99
F1	-35.778	10.54
F2	-33.21	7.5
F2	-32.12	7.2
G1	-36.72	12.25
G2	-29.12	8.2
G3	-41.02	14.25
G4	-42.46	14.45

Table 4: Signal to Noise Ratio and RDI

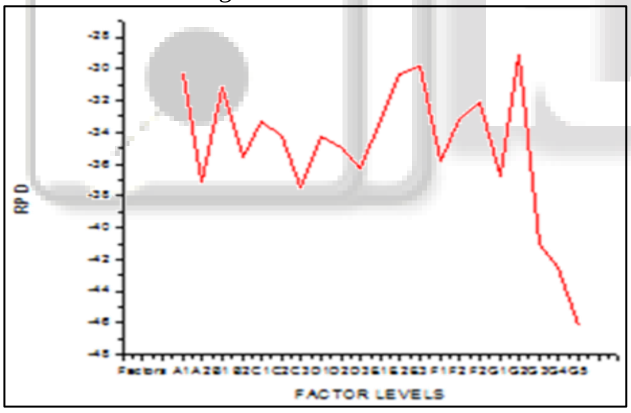

Fig. 3: signal to noise plot for each level factors

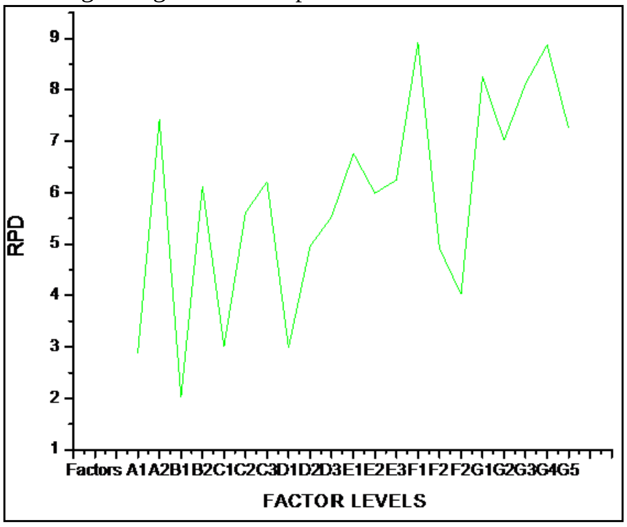

Fig. 4: RPD plots for each levels

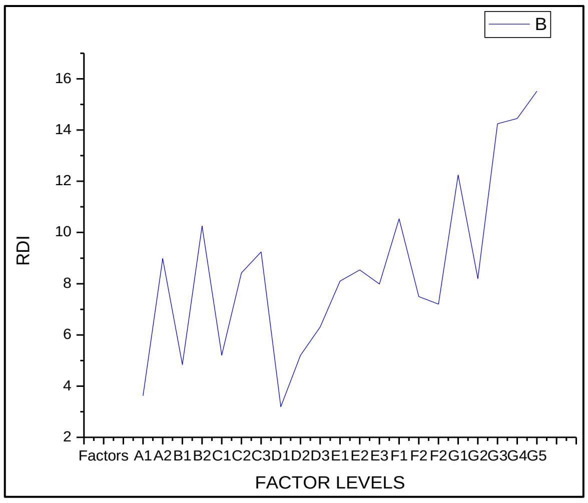

Fig. 5: RPD plots for each levels

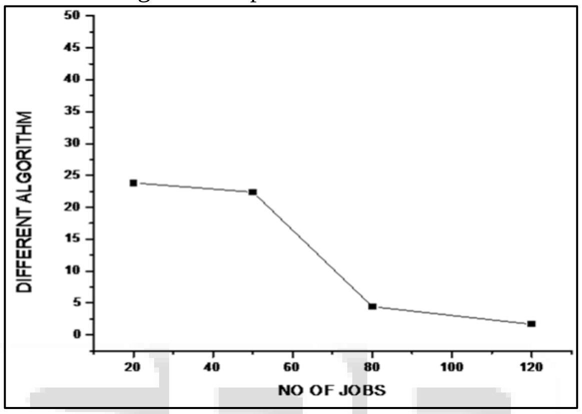

Fig. 6: different algorithms means plots

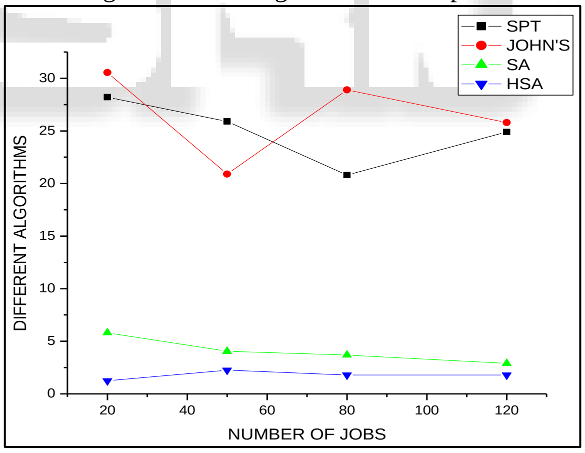

Fig. 7: 2 stage hybrid flow shop minimizing the makespan.

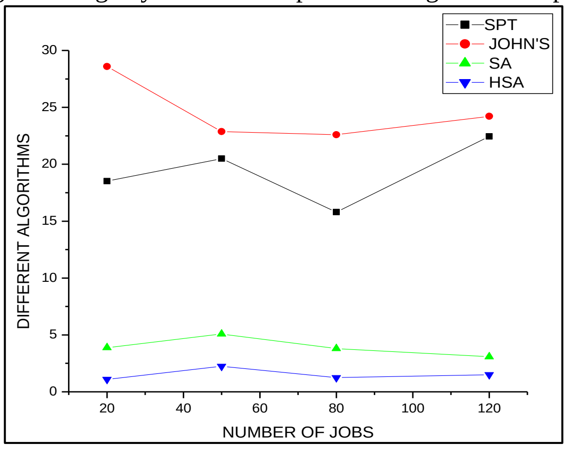

Fig. 8: 4 stage hybrid flow shop minimizing the make span.

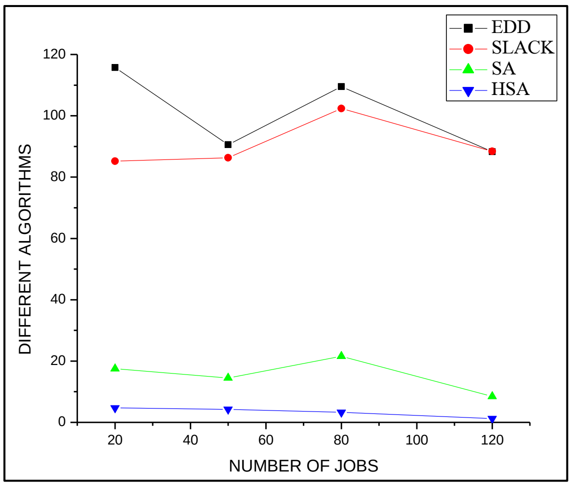

Fig. 9: 2 stage hybrid flow shop maximize tardiness.

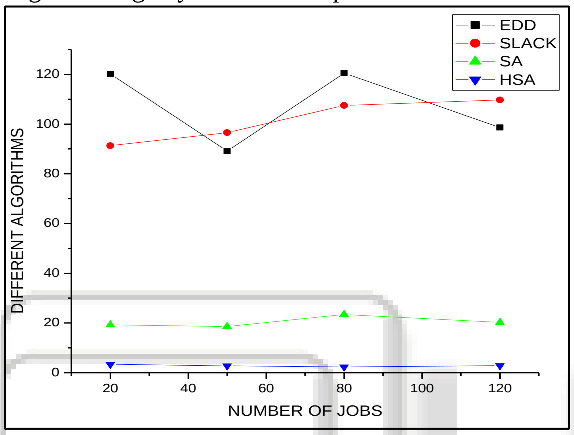

Fig. 10: 4 stage hybrid flow shop maximize tardiness.

The results of the n jobs m machines combinations see the graphs, HAS yield better output, RDI results with compared to other algorithms is better valve obtained. HAS is compared to some high performing metaheuristics showing high performances in the literature of the problem. To make the sure that the algorithms retains robustness even explored the effect of rise in the numbers of jobs on the performance of the algorithms.

The performance of HSA is constant as regards increasing the number of jobs while making comparison against other. As Fig depicts, when the number of jobs rises, the performance of SA improves, whereas other algorithms maintain their robustness.

V. Conclusions

In this project work the problem of hybrid flowshops scheduling with sequence-dependent setup times over two basically different objective functions, namely, makespan and maximum tardiness, was investigated. To tackle such an NP-hard problem, SA is hybridized with a simple local search to further equip the algorithm with a new strong tool to promote the quality of final solution of the proposed SA algorithm.

An exhaustive comparison between different operators and parameters of simulated annealing is conducted to obtain the precise calibration by means of the Taguchi method. In order to appraise the effectiveness of the proposed algorithm, compared the HSA against some high-performing metaheuristics is compared in the literature of the problem. Comparative analogy between the performances of the algorithms revealed the absolute superiority of the proposed simulated annealing. To make sure that the algorithm retains its robustness, the effect of rise in the number of jobs on the performances of the algorithms is also done.

VI. Future Work

As a direction for future work, it could be interesting to apply some other metaheuristics, such as electromagnetic-like algorithms and the adaptation of heuristic, and compare them with HSA or to examine the performance of the algorithm in some other complex scheduling problems, such as job shop and open shop, to see as to whether the high performance of the HAS is transferable to the other scheduling problems. Another clue for future work is the consideration of some other realistic assumptions such as machine availability constraints and unrelated machines. Another opportunity for project work is the consideration of the problem with the other optimization objectives such as minimization of total completion time, total tardiness, early and tardy penalties, job waiting time variance, or even multi-objective cases can be done.

References

[1] Zandieh et al., 2006 M. Zandieh, S.M.T. Fatemi Ghomi and S.M. Moattar Husseini, An immune algorithm approach to hybrid flowshops scheduling with sequence dependent setup times, *Journal of Applied Mathematics and Computation* 180 (2006), pp. 111–127.
[2] Ruiz and Stützle, 2008 R. Ruiz and T. Stützle, An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives, *European Journal of Operational Research* 187 (3) (2008), pp. 1143–1159.
[3] Ross, 1989 R.J. Ross, Taguchi techniques for quality engineering, McGraw-Hill, USA (1989).
[4] Cheng and Chang, 2007 B.W. Cheng and C.L. Chang, A study on flowshop scheduling problem combining Taguchi experimental design and genetic algorithm, *Expert Systems with Applications* 32 (2007), pp. 415–421.
[5] Gupta, 1988 J.N.D. Gupta, Two-stage hybrid flowshop scheduling problem, *Journal of the Operational Research Society* 39 (4) (1988), pp. 359–364
[6] Johnson, 1954 S.M. Johnson, Optimal two and three-stage production schedules with setup times included, *Naval Research Logistics Quarterly* 1 (1954), pp. 61–67.
[7] Kim, 1993 Y.D. Kim, Heuristics for flowshop scheduling problems minimizing mean tardiness, *Journal of Operational Research Society* 44 (1993), pp. 19–28
[8] Kurz and Askin, 2004 M.E. Kurz and R.G. Askin, Scheduling flexible flow lines with sequence-dependent setup times, *European Journal of Operational Research* 159 (1) (2004), pp. 66–82.
[9] Adenso-Dı´az, 1996 B. Adenso-Dı´az, An SA/TS mixture algorithm for the scheduling tardiness problem, *European Journal of Operational Research* 88 (1996), pp. 516–524.