# A Hybrid Algorithm for Job Shop Scheduling Problem

**Florentina Alina TOADER**

Petroleum Gas University of Ploiesti,
39 Bucuresti Blvd , Ploiesti, 100680, Romania
toader_florentina_alina@yahoo.om

**Abstract:** The Job Shop Scheduling Problem (JSSP) is regarded as one of the most challenging issues by the research community in this field due to its complexity. This paper presents a hybrid algorithm called H-PSO-SA for JSSP which is a mixture of two computational artificial intelligence algorithms: Particle Swarm Optimization and Simulated Annealing. In order to demonstrate efficiency of the proposed hybrid algorithm, a series of tests are conducted using a set of classical JSSP benchmarks. The schedule results are compared with outcomes well known in the scientific literature.

**Keywords:** Production Scheduling, Job Shop Scheduling Problem, Simulated Annealing, Particle Swarm Optimization

## 1. Introduction

The Job Shop Scheduling Problem (JSSP) represents one of the most difficult to solve problems in the industrial environment. This type of problem is categorized as NP-hard by Srinivas and Allada in 2004 [1]. Mesghouni et all mentioned in [2] that the complexity of this type of problem comes from the fact that consists of satisfying multiple goals. A set of resources needs to be allocated in a manner that satisfy the proposed goals, maximize the machine utilization and minimizes both the makespan and the total completion time of the entire process schedule.

The solution proposed most recently in the specific literature refers to the hybrid methods applied in this field that manages to combine the advantages of a local search technique (e.g. swarm intelligence which is based on collective intelligence models inspired by nature such as Ant Colony Optimization – ACO, Particle Swarm Optimization – PSO, Artificial Bee Colony –ABC, Wasp Behaviour Model – WBM, etc.) with the advantages of the global search techniques ( Simulated Annealing – SA, Tabu Search – TS, etc.).

This paper proposes a hybrid algorithm resulted by combining an artificial intelligence technique, Particle Swarm Optimization with a very popular metaheuristic, Simulated Annealing.

The goal of this research work is to identify an optimal solution for the JSSP by combining the strengths and minimizing the influence of each algorithm weak points.

## 2. State of the Art

JSSP is one of the most complex combinatorial optimization problems and the researcher's activity in this area is focused on finding an appropriate solution for it. There are various methods proposed for solving this problem, starting with traditional approaches based on mathematical or priority rules programming and ending with artificial intelligence and metaheuristic based methods [3].

During the recent years, a series of methods and algorithms specific to the artificial intelligence area were the target of the researchers' activity: Particle Swarm Optimization (PSO) [4], [14], [17], [23], Simulated Annealing (SA) [8], [16], Genetic Algorithms (GA) [15], [32], Tabu Search (TS) [18], etc.

However, at this moment the researchers' attention is focused on several hybrid algorithms that seem to be more suitable for real practical manufacturing environments implying larger sets of constraints, which are difficult to optimize by just using a simple metaheuristic.

Sha and Hsu introduced a hybrid algorithm based on PSO and Tabu Search [5] which proposes a different particle representation and modifies the particle movement thru the search space in order to obtain better results than traditional metaheuristic approach.

[6] introduces a hybrid algorithm based on Artificial Bee Colony Algorithm for flexible job shop scheduling problems with an external Pareto archive for partial solutions. The hybrid

algorithm presented in [10] proposes another different approach for multi-objective PSO for JSSP. In [12] JSSP is scrutinized from another angle, by using a combination between Genetic Algorithms (GA) and Simulated Annealing techniques. The simulated annealing process is implemented at each GA step as a mutation operation.

Another hybrid algorithm that involves Tabu Search and Ant Colony Optimization is proposed in [7] in order to obtain a JSSP optimal schedule.

A series of hybrid algorithms that combine the advantages of PSO and SA techniques were presented several recent articles. [9] introduces a hybrid algorithm, based on these JSSP techniques, that proposes a modification in the simulated annealing process: the new solution is searched in the neighbourhood of the original solution by using a PSO designed search. Another attempt of combining PSO and SA algorithms for periodic events scheduling in the context of JSSP is given in [22]. The hybrid algorithm shown in [24] proposes the construction of parallel initial solutions for SA using PSO in order to increase the SA efficiency. The global search capability of PSO is combined with three neighbourhood SA algorithms and tested on a set of classical benchmarks in [25]. The makespan minimization in the JSSP context represents an issue tackled in [26] by combining the efficiency of SA technique with a local search method based on PSO.

The above presented research works support the hypothesis that the application of a hybrid algorithm for the JSSP is suitable and efficient, contrasting to classical scheduling algorithms or simple artificial algorithms. As such, a plethora of different approaches for the hybrid algorithm implementation is available and gives a lot of space for further research.

The current paper pursues the author research work that is presented in [11], an article which introduced a study regarding the implementation in the JSSP field of two swarm intelligence techniques: Ant Colony Optimization and Particle Swarm Optimization. This related research revealed the fact that PSO manages to obtain the most suitable solutions for the proposed problem. The research is continued by proposing an amalgamation of the advantages of the local search technique represented by PSO with the ability to avoid the local optima trap proposed by SA method.

## 3. Problem Description

In the manufacturing systems area, the production scheduling represents one of the production management key points. The main objectives of the production scheduling are represented by maximizing the productivity and minimizing the attached costs.

Several types of manufacturing systems are staged in the real world. Taking into account their specifics and the involved production process complexity, a system classification is given by [13]: single stage system, single machine system, parallel machine system, multi-stage flow shop system and multi-stage job shop system.

The scheduling problem complexity increases from the single stage system to the multi-stage job shop system.

According to this classification, the Job Shop Scheduling Problem is among the hardest combinational problem in the manufacturing system field.

The mathematical model for the JSSP is described in [11] using the following notations:

- $m$ - number of machines;
- $n$ - number of product types;
- $M=\{M_1, M_2, \ldots, M_m\}$ - set of machines;
- $P=\{p_1, p_2, \ldots, p_n\}$ - a set of products;
- $S=\{s_1, s_2, \ldots, s_n\}$ - a set of series of each product;
- $n_i$ - number of operation corresponding to the product $p_i$;
- for each $p_i \in P$ is defined by a ordered set of operations $O_i$, an ordered set of corresponding machines $M_i$, and an ordered time values set attached to the specific operations $t_i$.

The problem solution is represented by the plan $\pi=(\mathrm{p}_\mathrm{i}, \mathrm{s}_\mathrm{i}, \pi_{ij})$, where [11]:

- $\pi_{ij}=(O_i, M_i, A_i, E_i)$
- $A_i=\{ta_1^i, ta_2^i, \ldots, ta_{n_i}^i\}$ the accessing time list on each machine;
- $E_i=\{te_1^i, te_2^i, \ldots, te_{n_i}^i\}$ the completion time list on each machine;

In order to better define the context of the problem, a set of assumptions have been made:

- One machine cannot substitute another;
- At a specific moment of time one single product can be allocated to a given machine;
- The operations are non-preemptive;
- The machines breakdown is not considered.

The main objectives of the schedule problem are: minimizing the starting time $t^i_j$ of each operation $o^i_j \in O_i$, minimizing the idle time $Id_i$ of each machine $M_i$, and the total completion time $C_{max}$. The objective function that evaluates each proposed schedule quality is introduced in the following equation (1).

$$f = \frac{1}{C_{max} + \sum_{i=1}^{n} Id_i + \sum_{i=1}^{n} \sum_{j=1}^{S_i} t_{ij}} \tag{1}$$

## 4. The Hybrid Algorithm

This section describes two selected methods (SA and PSO) and the algorithms adapted for the proposed problem. After that, a hybrid algorithm for solving JSSP, based on those methods, is presented and detailed, by showing the input parameters, the solution encoding, the structure of the output solution and the algorithm description.

The author decides to focus on presenting a new hybrid algorithm design for solving the proposed problem. The related literature presented above sustains the fact that this approach is more appropriate for real manufacturing environments which are difficult to optimize by just using a simple metaheuristic, as presented in [4], [8], [14], [15], [16], [17], [18], [23], [32].

### 4.1 Simulated Annealing

Simulated Annealing is a metaheuristic suitable for solving combinatorial optimization problems. This method is valued because it helps escaping from the local optimal trap and directs the final solution to the global optimal [20].

The notion of slow cooling specific to the annealing process in the metallurgy is implemented in SA algorithm as the slow probability diminution of accepting worse solutions in the process of exploring the search space [19].

The main steps of the SA algorithm adapted for JSSP, described by the mathematical model described in section 3, are the following:

- Step 1. Initialize the system temperature T=T$_0$;
- Step 2. Initialize the system cooling rate α
- Step 3. Randomly generate the initial schedule solution S
- Step 4. For *i←1: n* do
    1. Randomly generate schedule solution S* from the neighbourhood of S
    2. If f(S*)>f(S) then S=S*
    3. Reduce the temperature T = T*α
- Step 5. If the termination criteria is not satisfied repeat from Step. 4.

### 4.2 Particle swarm optimization

PSO is a technique specific to swarm intelligence inspired by the social comportment of beings living in large groups (such as flocks of birds or schools of fish), which uses the transmitting and sharing of the information between individuals [21].

The algorithm starts with a randomly initiated population of possible solutions. The search process is accomplished by a system of artificial particles that move through the search space with a certain speed calculated according to all the system characteristics.

The particle movement through the search space is influenced by two important indices: the best personal location of the current particle and the best global location of a particle in the past, as presented in the following equation (2) [21]:

$$\begin{aligned} v_{id} = v_{id} &+ \O_1 * rand()(poz_{id} - x_{id}) \\ &+ \O_2 * rand()(poz_{gd} - x_{id}) \end{aligned} \tag{2}$$

Where

- $X_i = (x_{i1}, \ldots, x_{id},)$ is an particle;
- $V_i = (V_{i_1}, \ldots, V_{i_{id}})$ is the particle speed;
- $Poz_i = (poz_{i_1}, \ldots poz_{i_{id}})$ is the best position of the particle;
- g is the index of the best global position;
- $\O_1$ and $\O_2$ are two constant values between 0 and 1: the cognitive parameter and the social parameter.

The main steps of the PSO algorithm adapted for JSSP, defined by the mathematical model described in section 3, are the following:

```
- Step 1. For i←1: number_of_particles do
      a) Initialize particle P_i
      b) Initialize pBest_P_i
- Step 2. Initialize globalBest
- Step 3. For i←1: number_of_particles do
      a) Calculate f(P);
      b) If f(P)>f(pBest) then pBest=P
      c) If f(S*)>f(S) then S=S*
- Step 4. Choose Gbest
- Step 5. For i←1: number_of_particles do
      a) Calculate particle P_i speed according to equation (2)
      b) Actualize particle P_i position according to equation (2)
- Step 6. If the termination criteria is not satisfied repeat from Step. 3.
- Step 7. Display gBest
```

## 4.3 Hybrid algorithm description

The H-PSO-SA hybrid algorithm merges the advantages of two optimizing methods for the JSSP and proposes the combination between the local search and global search in the solution space.

The idea of designing a hybrid algorithm based on PSO and SA techniques was successfully applied in different research papers in recent years (such as [9], [22], [24], [25], and [26]).

This paper proposes a new approach on the algorithm design, an approach illustrated by the pseudo code bellow. The algorithm performances are tested on a series of classical benchmarks and the solutions quality is evaluated according to the input data tests expected for optimal results.

### 4.3.1 Input parameters

The input data set for the H-PSO-SA is represented by:

- $m$ – number of available machines;
- $n$ – products number;
- $M=\{M_1, M_2, \ldots, M_3\}$ - machines list;
- $P=\{p_1, p_2, \ldots, p_n\}$ - product list;
- $s_i \in S$ – number of batches of each product $p_i, \forall p_i \in P$;
- $O_i=\{o_1^i, o_2^i, \ldots, o_{n_i}^i\}$ – an ordered set of operations that needs to be executed to obtain the finite product $p_i$;
- $M_i=\{m_1^i, m_2^i, \ldots, m_{n_i}^i\}$ – an ordered list of machine allocation for each operation;
- $TP_i=\{tp_1^i, tp_2^i, \ldots, tp_{n_i}^i\}$ – an ordered list of execution times attached to each operation;
- Np – number of available particles;
- $\varnothing_1 \in (0,1)$ – cognitive parameter;
- $\varnothing_2 \in (0,1)$ – social parameter;
- $I_{max}$ - maximum number of iteration for PSO algorithm;
- $T_0$ – initial temperature for SA algorithm;
- $T_{min}$ – minimum accepted temperature for SA algorithm;
- $\alpha$ – cooling rate for SA algorithm

### 4.3.2 Encoding scheme

In order to represent the candidate solution for JSSP, a permutation of jobs that is designed according to the mathematical model presented in section 3 is used.

In this context each candidate solution is structured as a production schedule $\pi=(p_i, s_i, \pi_{ij})$ defined by a plan $\pi_{ij}=(O_i, M_i, A_i, E_i),\ \forall i \in \overline{1:n}$.

A complex function is designed in order to verify and approve the proposed schedule by taking into consideration the system structure and the production demands. If any discrepancies appear, the necessary modifications are made in order to obtain a reliable and feasible schedule solution.

### 4.3.3 Output solution

The output data set for H-PSO-SA algorithm is represented by the production schedule with the best value for the attached objective function.

Since this objective function is calculated taking into consideration the starting time $t_j^i$ of each operation $o_j^i \in O_i$, the idle time $Id_i$ of each machine $M_i$ and the total completion time $C_{max}$ and taking into account the equation (1), it is obvious that the value needs to be maximized. So the higher the $f$ function value is, the higher the quality of the attached production schedule is.

### 4.3.4 H-PSO-SA algorithm

The proposed H-PSO-SA algorithm includes two important phases:

1. The initial particle population is not generated by following the traditional routine. For each initial particle a random schedule is generated, passed through an SA algorithm and after that included in the initial population.
2. An algorithm composed by combined methods of PSO and SA is executed.

The H-PSO-SA pseudo code is presented:

```
Step 1. Read input data set
Step 2. For $i \leftarrow 1$: Np do
    a) Randomly initialize particle $P_i$
    b) Approve($P_i$)
    c) Initialize the system temperature $T=T_0$;
    d) Initialize the system cooling rate $\alpha$
    e) While T>Tmin do
        i. Randomly generate particle $P_i$* from the neighbourhood of $P_i$
        ii. Approve($P_i$*)
        iii.
        iv. If f($P_i$*)>f($P_i$) then $P_i$=$P_i$*
        v. $T = T*\alpha$
    f) Initialize pBest_$P_i$
Step 3. Initialize globalBest
Step 4. Iteration=0
Step 5. While Iteration< $I_{max}$ do
    a) For $i \leftarrow 1$: Np do
        i. Randomly initialize particle $P_i$
        ii. Approve($P_i$)
        iii. Initialize the system temperature $T=T_0$;
        iv. Initialize the system cooling rate $\alpha$
        v. While T>Tmin do
            1. Randomly generate particle $P_i$* from the neighbourhood of $P_i$
            2. Approve($P_i$*)
            3. If f($P_i$*)>f($P_i$) then $P_i$=$P_i$*
            4. $T = T*\alpha$
        vi. If f($P_i$) > f(pBest) then
            1. f( pBest)=f($P_i$)
            2. $\pi_{pBest}=\pi_{p_i}$
    b) For i←1: Np do
        i. If f(gBest) >f($P_i$) then
            1. gBest=$P_i$
            2. $\pi_{pBest}=\pi_{p_i}$
    c) For P=1, $Np$ do
        i. $v_p=v_p+rand(\varnothing_1)*\pi_{gBest} + rand(\varnothing_2)*\pi_{pBest}$
        ii. $\pi_p=\pi_p+v_p$
    d) Iteration++
Step 6. Display Solution
```

### 4.3.5 Algorithm validation.

Because there are no guarantees that the proposed H-PSO-SA is able to identify an optimal solution for the JSSP, tests are made by applying a set of 15 classical, well known in the scientific literature benchmarks, presented in Table 1. The results of the hybrid algorithm execution will be compared with the ones available for the considered benchmarks.

**Table 1.** Input benchmarks

| **Benchmark Name** | **Dimension (Machines x Jobs)** | **Scientific Literature Reference** |
|---|---|---|
| ABZ5 | 10X10 | [27] |
| ABZ6 | 10X10 | [27] |
| FT6 / MT06 | 6X6 | [28] |
| FT10 / MT10 | 10X10 | [28] |
| LA01 | 10X5 | [29] |
| LA02 | 10X5 | [29] |
| LA03 | 10X5 | [29] |
| LA04 | 10X5 | [29] |
| LA19 | 10X10 | [29] |
| LA20 | 10X10 | [29] |
| ORB1 | 10X10 | [30] |
| ORB2 | 10X10 | [30] |
| ORB3 | 10X10 | [30] |
| ORB4 | 10X10 | [30] |
| ORB5 | 10X10 | [30] |

## 5. Experimental Results

The performance of the hybrid algorithm was evaluated for 15 input classical benchmarks (the ones used most often in the related literature), as presented in the first table. 12 input data test parameters specific for the two initial algorithms (that were the basic technique upon which the hybrid algorithm is build) were considered.

**Table 2.** Input parameters sets for H-PSO-SA

| **Parameters set name** | **Np** | $\varnothing_1$ | $\varnothing_2$ | $I_{max}$ | $T_0$ | $T_{min}$ | $\alpha$ |
|---|---|---|---|---|---|---|---|
| PS1 | 50 | 0.2 | 0.2 | 50 | 25 | 0.01 | 0.1 |
| PS2 | 50 | 0.2 | 0.4 | 50 | 25 | 0.01 | 0.1 |
| PS3 | 50 | 0.4 | 0.2 | 50 | 25 | 0.01 | 0.1 |
| PS4 | 50 | 0.4 | 0.4 | 50 | 25 | 0.01 | 0.1 |
| PS5 | 50 | 0.4 | 0.4 | 80 | 25 | 0.01 | 0.1 |
| PS6 | 50 | 0.6 | 0.4 | 50 | 25 | 0.01 | 0.1 |
| PS7 | 50 | 0.4 | 0.6 | 50 | 25 | 0.01 | 0.1 |
| PS8 | 50 | 0.6 | 0.6 | 50 | 25 | 0.01 | 0.1 |
| PS9 | 50 | 0.6 | 0.8 | 80 | 25 | 0.01 | 0.1 |
| PS10 | 50 | 0.8 | 0.6 | 80 | 25 | 0.01 | 0.1 |
| PS11 | 100 | 0.2 | 0.2 | 50 | 20 | 0.01 | 0.1 |
| PS12 | 100 | 0.2 | 0.4 | 50 | 20 | 0.01 | 0.1 |
| PS13 | 100 | 0.4 | 0.2 | 50 | 20 | 0.01 | 0.1 |
| PS14 | 100 | 0.4 | 0.4 | 50 | 20 | 0.01 | 0.1 |
| PS15 | 100 | 0.4 | 0.4 | 80 | 20 | 0.01 | 0.1 |
| PS16 | 100 | 0.6 | 0.4 | 50 | 20 | 0.01 | 0.1 |
| PS17 | 100 | 0.4 | 0.6 | 50 | 20 | 0.01 | 0.1 |
| PS18 | 100 | 0.6 | 0.6 | 50 | 20 | 0.01 | 0.1 |
| PS19 | 100 | 0.6 | 0.8 | 80 | 20 | 0.01 | 0.1 |
| PS20 | 100 | 0.8 | 0.6 | 80 | 20 | 0.01 | 0.1 |

Table 2 contains the input data sets for the parameters that are specific to the hybrid algorithm: Np (number of available particles), $\varnothing_1$ (cognitive parameter), $\varnothing_2$ (social parameter), $I_{max}$ (maximum number of iteration for PSO algorithm), $T_0$ (initial temperature for SA algorithm), $T_{min}$ (minimum accepted temperature for SA algorithm), $\alpha$ (cooling rate for SA algorithm). In order to test the proposed algorithm performance in various situations, 20 input data sets for these parameters, with very different values, are taken into consideration.

Table 3 presents the experimental results obtained after performing the entire test on the proposed algorithm.

**Table 3.** Experimental results

| **Benchmark Name** | **Lower Bound** | **Upper Bound** | **Optimal** | **H-PSO-SA optimal** |
|---|---|---|---|---|
| ABZ5 | 868 | 1370 | **1234** | **1234** |
| ABZ6 | 742 | 1071 | 943 | 948 |
| FT6 / MT06 | 46 | 68 | **55** | **55** |
| FT10 / MT10 | 655 | 1262 | **930** | **930** |
| LA01 | 666 | 830 | 666 | 671 |
| LA02 | 635 | 844 | **655** | **655** |
| LA03 | 588 | 773 | 597 | 598 |
| LA04 | 537 | 839 | **590** | **590** |
| LA19 | 685 | 1046 | **842** | **842** |
| LA20 | 756 | 1210 | **902** | **902** |
| ORB1 | 695 | 1456 | **1059** | **1059** |
| ORB2 | 671 | 1157 | **888** | **888** |
| ORB3 | 648 | 1297 | **1005** | **1005** |
| ORB4 | 759 | 1356 | 1005 | 1020 |
| ORB5 | 630 | 1116 | **887** | **887** |

The H-PSI-SA optimal column represents the total completion time returned by the hybrid algorithm for the proposed problem. This result is compared with the known lower bound and upper bound for the same problem, and also with the optimal value proposed by the scientific literature [31].

All the discussed experiments have been made on a desktop computer with the following hardware configuration: AMD FX™-6100 Six-Core Processor, 3.30 GHz CPU.

The software tool used for the simulation was QTCreator version 2.6.2, which is able to run programs written in the C++ programming language.

The experimental results presented above highlight the fact that the hybrid algorithm H-PSO-SA outcomes (regarding the total completion time) manage to fit into the interval between the lower bound and the upper bound described in the specific literature. Furthermore, in 73.33% of the considered cases, the hybrid algorithm manages to achieve a solution that is equal with the optimal value of the makespan presented in the scientific literature.

Figure 1 presents the graphical representation of the experimental results detailed in table 3 and highlights the performances of the hybrid algorithm solutions.

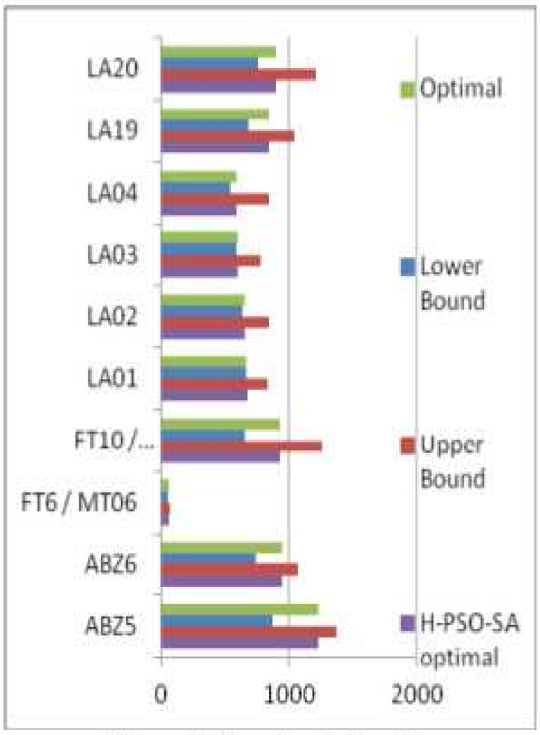

**Figure 1.** Experimental results

Table 4 presents the experimental results that measure the algorithms performance, considering the total running time (presented in milliseconds), the name of the input set, and the name of specific parameters that lead to the optimal solution.

According to these results, the proposed hybrid algorithm usually reaches an optimal solution for a higher number of particles (80% of the considered cases) and with a lower starts temperature. Also the high values for the cognitive and social parameter seems to be favourable for achieving a better solution.

The experimental results presented above sustain the fact that the hybrid algorithm H-PSO-SA is suitable for solving the JSSP. The algorithm manages to obtain solutions that are equivalent to the best solution shown so far in the specific literature for almost all the classical benchmarks tested.

**Table 4.** Experimental results

| Benchmark Name | H-PSO-SA optimal | Running time (milliseconds) | H-PSO-SA parameter set |
|---|---|---|---|
| ABZ5 | 1234 | 87458 | PS18 |
| ABZ6 | 948 | 94658 | PS16 |
| FT6 / MT06 | 55 | 9296 | PS16 |
| FT10 / MT10 | 930 | 145301 | PS18 |
| LA01 | 671 | 10896 | PS16 |
| LA02 | 655 | 10122 | PS18 |
| LA03 | 598 | 11765 | PS7 |
| LA04 | 590 | 11021 | PS8 |
| LA19 | 842 | 10270 | PS16 |
| LA20 | 902 | 11891 | PS18 |
| ORB1 | 1059 | 138536 | PS18 |
| ORB2 | 888 | 124744 | PS8 |
| ORB3 | 1005 | 133435 | PS18 |
| ORB4 | 1020 | 137829 | PS16 |
| ORB5 | 887 | 125469 | PS16 |

In comparison with other scientific papers presented in related literature section, the proposed hybrid algorithm manages to obtain better results for the tested input data, regarding the total completion time that in 73.33% of the cases is equal to the known optimal. This accomplishment is due to the complex formulation of the fitness function that manages to guide the algorithm into the right search direction for optimal solution.

One disadvantage of the proposed algorithm refers to the fact that the running time has high values.

Even under this disadvantage, the experimental results presented in this paper show that the proposed hybrid algorithm can be successfully used in solving the complex problem of JSSP.

## 6. Conclusions and Future Works

Considering the impact that an optimal production planning has for the production management quality, in recent years the researchers attention focused on finding algorithms that satisfy this demand.

The Job Shop Scheduling represents one on the most complex and challenging problems in the manufacturing system field. This paper reveals a new approach for solving this difficult problem consisting in a hybrid algorithm H-PSO-SA based on two artificial intelligence techniques: Particle Swarm Optimization and Simulated Annealing. The proposed H-PSO-SA is tested for 15 classical benchmarks which are available in the scientific literature and for 20 different input data tests consisting of different values for the specific algorithm parameters.

The comparison results and discussions reveals the fact that the hybrid algorithm H-PSO-SA is suitable for solving the proposed problem and manages to obtain a value equal with the optimal makespan given by the scientific literature in 73.33% of the considered cases, by combining the PSO and SA algorithms strengths and minimizing the influence of each algorithm weak points. In the other cases the makespan value returned by the hybrid algorithm is not considerably higher than the optimal value and manages to fit into the interval defined by the lower and upper bound. The future research will focus on finding solutions for optimizing the hybrid algorithm, in order to minimize the total running time and to test the proposed algorithm for larger size benchmarks. Another future challenge is represented by taking into consideration the random machines malfunctions and testing the algorithm performance in this context too.

## REFERENCES

1. SRINIVAS, M. K. T., V. ALLADA, **Solving the Machine Loading Problem in a Flexible Manufacturing System using a Combinatorial Auction-based Approach**, International Journal of Production Research, vol. 42, no. 9, 2004, pp. 1879-1893.

2. MESGHOUNI, K., S. HAMMADI, P. BONE, **Evolutionary Algorithms for Job Shop Scheduling,** International Journal of Applied Mathematics and Computer Science, Vol. 14, No.1, 2004, pp. 91-103.

3. HOLLAND, J. H., **Adaptation in Natural and Artificial Systems,** MIT Press, Cambridge, 1992

4. SHA, D. Y., H. H. LIN, **A multi-objective PSO for Job Shop Scheduling Problems,** Expert Systems with Applications, Vol. 37, Issue 2, 2010, pp. 1065-1070.

5. SHA, D. Y., C. Y. HSU, **A Hybrid Particle Swarm Optimization Approach for the Job Shop Scheduling Problem**, Computer and Industrial Engineering, Vol.51, Issue 4, 2006, pp. 791-808.

6. LI, J., Q. PAN, S. XIE, S. WANG, **A Hybrid Artificial Bee Colony Algorithm for Flexible Job Shop Scheduling Problem,** International Journal of Computers, Communications & Control, Vol. IV, No. 2, 2011, pp. 286-296.

7. ESWARAMURTHY, V. P., A. TAMILARASI, **Hybridizing Tabu Search with Ant Colony Optimization for Solving Job Shop Scheduling Problems,** International Journal of Advanced Manufacturing Technology, Vol.40, Issue 9-10, 2010, pp. 1004-1015.

8. BOZEJKO, W., J. PEMPERA, C. SMUNTNICKI, **Parallel Simulated Annealing for the Job Shop Scheduling Problem,** Lecture notes in computer science, Proceedings of the 9th International Conference on Computational Science, Vol. 5544, 2009, pp. 631-640.

9. GE, H., W. DU, F. QIAN, **A Hybrid Algorithm Based on Particle Swarm Optimization and Simulated Annealing for Job shop Scheduling,** Proceedings of the Third International Conference on Natural Computation, Vol. 3, 2007, pp. 715-719

10. SHEN, J., F. YANO, T. SHOHDOHJI, Y. TOYODA, **An Approach to Multi-objective Job Shop Scheduling using Hybrid Particle Swarm Optimization,** Proceedings of the 38th International Conference on Computers and Industrial Engineering, Beijing, Republic of China, 2008, pp. 1836-1843.

11. TOADER, F. A., **Production Scheduling Using ACO and PSO Techniques,** Proceedings of the International Conference on Development and Applied Systems, 2014, pp. 170-175.

12. TAMILARASI, A., T. ANANTHA KUMAR, **An Enhanced Genetic Algorithm with Simulated Annealing for Job Shop Scheduling,** International Journal of Engineering, Science and Technology, Vol. 2, No. 1, 2010, pp. 144-151.

13. SAWIK, T., **Production Planning Scheduling in Flexible Assembly Systems**, Springer, 1999

14. PONGCHAIRERKS, P., V. KACHITVICHYANUKUL, **A Particle Swarm Optimization Algorithm on Job Shop Scheduling Problem with Multi-Purpose Machine**, Asia-Pacific Journal of Operational Research, Vol. 2, Issue 2, 2009, pp.161-184.

15. ZOBOLAS, C. I., C. D. TRANTILIS, G. IOANNOU, **A Hybrid Evolutionary Algorithm for the Job Shop Scheduling Problem,** Journal of the Operational Research Society, Vol. 60, No. 2, 2009, pp. 221-235.

16. ZHANG, R., WU, C., **A simulated annealing algorithm based on block properties for the job shop scheduling problem with total weighted tardiness objective**, Computers & Operations Research, Vol. 38, No. 5, 2011, pp. 854-867.

17. ZHANG, R., S. SONG, C. WU, **A Two-stage Hybrid Particle Swarm Optimization Algorithm for the Stochastic Job Shop Scheduling Problem,** Knowledge Based Systems, Vol. 27, 2012, pp. 393-206

18. ZHU, Z. C., K. M. NG, H. L. ONG, **A Modified Tabu Search Algorithm for Cost-based Job Shop Problem,** Journal of the Operational Research Society, Vol. 61, No. 4, 2010, pp. 611-619.

19. MICHALEWICZ, Z, D. B. FOGEL, **How to Solve It: Modern Heuristics,** Springer, 2004.

20. VAN LAARHOVEN, P. J., E. H. AARTS, **Simulated Annealing: Theory and Application**, Kluwer Academic Publishing, 1992.

21. BONABEAU, E., M. DORIGO, T. THERAULAZ, **From Natural to Artificial Swarm Intelligence**, Oxford University Press, New York, 1999.

22. JAMILI, A., M. A. SHAFIA, R. TAVAKKOLI-MOGHADDAM, **A Hybrid Algorithm based on Particle Swarm Optimization and Simulated Annealing for Periodic Job Shop Scheduling Problem,** The International Journal of Advanced Manufacturing Technology, ISSN 0268-3768, vol. 54 (1-4), 2010, pp. 309-322.

23. BOUKEF, H., M. BENREJEB, P. BORNE, **Flexible Job-shop Scheduling Problems Resolution Inspired from Particle Swarm Optimization**, Studies in Informatics and Control, ISSN 1220-1766, vol. 17 (3), 2008, pp. 241-252.

24. SONG, X., Y. CAO, C. CHANG, **A Hybrid Algorithm of PSO and SA for Solving JSP,** Proceedings of the Fifth International Conference on Fuzzy Systems and Knowledge Discovery, ISBN 978-0-7695-3305-6, vol. 1, pp. 111-115.

25. SONG, C. L., X. B. LIU, W. WANG, B. XIN, **A Hybrid Particle Swarm Optimization Algorithm for Job Shop Scheduling Problem**, International Journal of Advancements in Computing Technology, vol. 3 (4), pp. 79-88.

26. WEIJUN, X., W. ZHIMING, Y. GENKE, **A New Hybrid Optimization Algorithm for the Job Shop Scheduling Problem**, Proceedings of the 2004 American Control Conference, ISBN 0-7803-8335-4, vol. 6, p. 5552-5557.

27. ADAMS, J., E. BALAS, D. ZAWACK, **The Shifting Bottleneck Procedure for Job Shop Scheduling**, Management Science vol. 34 (3), 1988, pp. 391-401.

28. FISHER, H., G. L. THOMPSON, **Probabilistic Learning Combinations of Local Job-shop Scheduling Rules,** J. F. Muth, G. L. Thompson (eds.), Industrial Scheduling, Prentice Hall, Englewood Cliffs, New Jersey, 1963, pp. 225-251.

29. LAWRENCE, S., **Resource Constrained Project Scheduling: an Experimental Investigation of Heuristic Scheduling Techniques (Supplement),** Graduate School of Industrial Administration,

Carnegie-Mellon University, Pittsburgh, Pennsylvania, 1984

30. APPLEGATE, D., W. COOK, **A Computational Study of the Job-shop Scheduling Instance**, ORSA Journal on Computing, vol. 3, 1991, pp. 149-156.

31. Online document. **CSP2SAT: JSS benchmark results**, http://bach.istc.kobe-u.ac.jp/csp2sat/jss/, cited at 18.02.2015.

32. NICOARA, E. S., F. G. FILIP, N. PARASCHIV, **Simulation-based Optimization using Genetic Algorithms for multi-objective Flexible JSSP**, Studies in Informatics and Control, ISSN 1220-1766, vol. 20 (4), 2011, pp. 333-344.