Evolutionary Algorithms for Job Shop Scheduling

Florentina Alina Toader  
Dept. of Informatics, Information Technology, Mathematics and Physics  
Petroleum Gas University of Ploiesti  
Ploiesti, Romania  
toader_florentina_alina@yahoo.com

Abstract – Job Shop Scheduling Problem (JSSP) represents a real challenge for the researchers’ community due to its complexity consisting in the plurality of resources that need to be optimally used and the variety of goals that needs to be accomplished. This paper presents the implementation of three Evolutionary Algorithms ( Genetic Algorithms, Particle Swarm Optimization and Ant Colony Optimization) for the JSSP. The tests are made considered a set of classical benchmarks for the proposed problem and the obtained results are subject to comparison.

Keywords – Job Shop Scheduling, Evolutionary Algorithms, Scheduling Optimization

I. INTRODUCTION

Scheduling problems consist of describing the production stages, taking into account the available raw materials and machines, the type of products and operating time on each machine without requiring continuous human operator intervention.

The JSSP was listed as problems with a high degree of complexity [1], mainly due to the multitude of restrictions that must be satisfied simultaneously (resources allocated to meet the global scope of the problem, minimizing the total production allocated time, maximize the machine utilization, etc.) [34].

The modern research is directed towards adopting evolutionary algorithms solutions (such as Ant Colony Optimization – ACO, Particle Swarm Optimization – PSO, Artificial Bee Colony – ABC, Genetic Algorithms – GA) and implementing them in this field [2].

In [3] Turguner and Sahingoz present a computer simulation study on the application of ACO solution for JSSP. The parallel machine scheduling problem is reached in [4], and ACO is used to minimize the delay costs and to increase the scheduling efficiency. Another approach for solving JSSP by using ACO is presented in [5] by limiting the trail intensities and focusing on searching the neighborhood of the best found solution. Other research papers that analyses the application of ACO in the JSSP field are [6], [7], [8].

In [9] is presented an investigation over the possibility of PSO implementation in JSSP field. The efficiency of PSO in production maximization for JSSP is discussed in [10] over a set of randomly generated instances. Mouiri et all proposed in [11] an effective PSO for solving flexible JSSP. The evaluation of the algorithms performance is made considering a set of classical benchmarks instances from the literature.

A multi-objective optimization solution for JSSP based on PSO technique is introduced in [12] by proposing a modified particle position representation and a different way of moving thru the search space. The simple and flexible job shop scheduling problem is also solved in [13] by using PSO. The main objective is to minimize the maximum completion time of all proposed jobs.

In [14] is introduced a GA approach over the optimization procedure for a flexible JSSP and the experimental results suggest the applicability of GA method in that area. In 2008 Guo et al introduce an optimization model based on GA for scheduling flexible assembly lines [15]. In [16] is presented a comparison of direct and indirect implementation types of GA in the static JSSP area and it is highlighted the fact that GA offers optimal solutions only for a part of the given experiments.

Gao et al introduced in 2015 [17] a two stage Artificial Bee Colony Algorithm for solving the JSSP by scheduling and re-scheduling when new jobs arrive in queue. In [18] and [19] is presented an improved ABC algorithm for JSSP that proposes the use of a mutation operation for a random solution in order to obtain better results. A survey over the metaheuristic algorithms approaches for solving flexible JSSP is detailed in [20].

This paper proposes an analysis over three Evolutionary Algorithms ( Genetic Algorithm, Particle Swarm Optimization and Ant Colony Optimization) implemented to solve JSSP. The tests are made considering a set of 15 classical input benchmarks and 45 sets of specific input parameters.

The results are compared and discussed in order to identify if the proposed algorithms are suitable to solve JSSP. Since optimally solving JSSP continue to be a challenge, it is important to view the algorithms’ behavior in this context and to identify the advantages and disadvantages of these methods in this area.
II. JOB SHOP SCHEDULING PROBLEM

Job Shop Scheduling Problem is recognized in the scientific literature as one of the most challenging and it was included in the NP-hard category in 2004 by Srivas and Allada [21]. The solution needs to provide a schedule that maximizes the productivity and minimizes the makespan and the production costs. The difficulty of finding an optimal solution for this problem increases as the system described is more complex.

The mathematical model used to describe JSSP is extensively described in [22] and uses the following notations:

- \( M = \{M_1, M_2, ..., M_m\} \) - the machine set, where \( m \) represents the number of available machines;
- \( P = \{p_1, p_2, ..., p_n\} \) - the product set, where \( n \) represents the number of available machines;
- \( S = \{s_1, s_2, ..., s_n\} \) - the product series;
- for each \( P_i \subseteq P \) ordered set of operations, \( O_{ij} \), is defined, an ordered set of corresponding machines \( M_i \), and an ordered time values set attached to the specific operations \( t_i \).

The JSSP solution is represented by the plan \( \pi_i = (P_i, S_i, \pi_{ij}) \).

The notation \( \pi_{ij} \) defines a set of 4 elements \((O_{ij}, M_i, A_i, E_i)\) where [23]:

- \( A_i = \{t_{a1}^i, t_{a2}^i, ..., t_{an}^i\} \) - the accessing time list on each machine, where \( n_i \) - number of operation corresponding to the product \( P_i \);
- \( E_i = \{t_{e1}^i, t_{e2}^i, ..., t_{en}^i\} \) - the completion time list on each machine;

The proposed solution is evaluated considering the satisfaction degree of the JSSP main objectives (minimizing the starting time \( t_{ij}^f \) and the idle time \( ld_{ij} \) of each operation \( o_{ij} \) \( \in O_i \), minimizing the total completion time \( C_{max} \)). Those goals are included in an objective function \( f \) introduced in equation (1) that evaluates the solution quality. The goal is to minimize the value of the \( f \) function.

\[
f = \frac{1}{C_{max} + \sum_{i=1}^{m} ld_{ij} + \sum_{i=1}^{m} \sum_{j=1}^{n} t_{ij}} \tag{1}
\]

III. EVOLUTIONARY ALGORITHMS

Evolutionary Algorithms (EA) represent a subset of Artificial Intelligence inspired by biological evolution that consists in a population-based metaheuristic optimization. The specific EA techniques could give the following: Genetic Algorithms, Particle Swarm Optimization, Ant Colony Optimization, Artificial Bee Colony, Cuckoo Search and other.

A. Genetic Algorithms

Genetic Algorithms represents a heuristic search method based on the natural evolution process. The Charles Darwin “survival of the fittest” principle represents the synthesizing idea of GA [24].

The GA population consists in a set of individuals that represent possible solutions. At each step of the evolution, the best individuals are selected for undergoing the crossover procedure in order to obtain new individuals that are added to the population. Also with a low probability a set of the new individuals are selected and the mutation operator is applied in order to introduce random minor modifications.

The specific input parameters for AG are:

- \( G_{max} \) represents the maximum number of generations;
- \( n_0 \) represents the number of individuals in first generation;
- \( n_{max} \) represents the maximum number of individuals;
- \( pc \in (0,1) \) is the crossover probability;
- \( pm \in (0,1) \) is the mutation probability;

To those are added the specific JSSP parameters, as they are presented in section II.

Each individual that represents a possible solution will be coded as a plan \( \pi_{64} = (P, S, \pi_{ij}) \) and his performance will be evaluated considering the value of the fitness function introduced in equation (1). The AG solution will be represented at the end of the evolution by the individual with the minimum fitness function.

B. Particle Swarm Optimization

Particle Swarm Optimization is a stochastic optimization technique inspired by the social behavior of birds’ flocks or school of fish. The particularity of this technique is represented by the transmission and sharing of information [30].

The particles are initialized with a set of pseudo-random candidate solutions and the optimum is searched by updating generations [31]. Individual’s movement thru the search space is influenced by the current global optimum and by the personal memory. Thus every particle has an adaptive speed that directs the movement and remembers the personal best position encountered so far.

The mathematical formulas that guide the particle movement thru the search space are presented in equations (2) and (3) [32].

\[
v_{id} = v_{id} + \alpha \cdot rand(1) (p_{o id} - x_{id}) + \beta \cdot rand(1) (p_{g id} - x_{id})
\]
where:

- $X_i = (x_{id_1}, \ldots, x_{id_n})$ represents a particle;
- $V_i = (v_{id_1}, \ldots, v_{id_n})$ represents the particle speed;
- $\text{POZ}_i = (\text{POZ}_{id_1}, \ldots, \text{POZ}_{id_n})$ represents the best visited position;
- $\mathcal{G}$ is the best global position;
- $\varnothing_1$ is the cognitive parameter;
- $\varnothing_2$ is the social parameter.

The specific input parameters for PSO are:

- $N_p$ represents the number of particles;
- $\varnothing_1$ is the cognitive parameter;
- $\varnothing_2$ is the social parameter.

To those specific parameters are added the specific JSSP parameters, as they are presented in section II.

Each particle that represents a possible solution will be coded as a plan $\pi_{PSO} = (P_i, S_i, H_i)$ and its performance will be evaluated considering the value of the fitness function introduced in equation (1). The particle with the minimum fitness function at the end of the simulation will represent the algorithm solution.

C. Ant Colony Optimization

The metaheuristic are inspired most often by natural processes. Ant Colony Optimization is a technique developed based on the ant colonies natural optimization techniques [33].

ACO algorithm is based on three base principles [32]:

- A set of formal asynchronous agents called ants move thru the search space states that correspond to partial solutions;
- With every move the solution is built incrementally by taking into account two parameters: pheromone paths and attractiveness;
- The ant modifies the pheromone path that will be used by the other ants in the future according to the current solution quality.

The ant probability of accepting to move from one state to another depends on the combination of two values: the attractiveness (defined as the overall probability of moving to a new state) and the pheromone trail (that shows how effective were the current movement in the past).

The specific input parameters for ACO are:

- $N_a$ represents the number of ants;
- $e_r \in (0,1]$ represents the pheromone evaporation rate;
- $\alpha \in (0,1]$ represents the pheromone trail influence.

Each ant represents a possible solution and is coded as a plan $\pi_{ACO} = (P_i, S_i, H_i)$. Its performance will be evaluated considering the value of the fitness function introduced in equation (1). The algorithms solution is represented by the ant with the minimum fitness function value.

IV. EXPERIMENTAL RESULTS

In order to test the selected Evolutionary Algorithms behavior in JSSP context, a set of classical benchmarks are taken into consideration. These benchmarks are presented in Table I, together with the problem dimension (number of machines and jobs) and the scientific literature source that is cited.

<table>
<thead>
<tr>
<th>Benchmark Label</th>
<th>Problem Dimensions (Machines X Jobs)</th>
<th>Cited Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABZ5</td>
<td>10X10</td>
<td>[25]</td>
</tr>
<tr>
<td>ABZ6</td>
<td>10X10</td>
<td>[23]</td>
</tr>
<tr>
<td>FT6 / MT06</td>
<td>6X6</td>
<td>[26]</td>
</tr>
<tr>
<td>FT10 / MT10</td>
<td>10X10</td>
<td>[28]</td>
</tr>
<tr>
<td>LA01</td>
<td>10X5</td>
<td>[27]</td>
</tr>
<tr>
<td>LA02</td>
<td>10X5</td>
<td>[27]</td>
</tr>
<tr>
<td>LA03</td>
<td>10X5</td>
<td>[27]</td>
</tr>
<tr>
<td>LA04</td>
<td>10X5</td>
<td>[27]</td>
</tr>
<tr>
<td>LA19</td>
<td>10X10</td>
<td>[27]</td>
</tr>
<tr>
<td>LA20</td>
<td>10X10</td>
<td>[27]</td>
</tr>
<tr>
<td>ORB1</td>
<td>10X10</td>
<td>[28]</td>
</tr>
<tr>
<td>ORB2</td>
<td>10X10</td>
<td>[28]</td>
</tr>
<tr>
<td>ORB3</td>
<td>10X10</td>
<td>[28]</td>
</tr>
<tr>
<td>ORB4</td>
<td>10X10</td>
<td>[28]</td>
</tr>
<tr>
<td>ORB5</td>
<td>10X10</td>
<td>[28]</td>
</tr>
</tbody>
</table>

The considered algorithms performance is tested using 15 input data sets for each one specific parameter, as they were described in section II.

In Table II are introduced the parameters for PSO. For example in the parameter set labeled DS_PSO_1 the number of particles considered is 50, the value considered for tests for the cognitive parameter is 0.4 and the test value for the social parameter is 0.4. In order to determine the optimal value for these parameters different tests are made with Np values between 50 and 100, $\varnothing_1$ values between 0.2 and 0.8 and $\varnothing_2$ values between 0.2 and 0.8.

Table III presents the parameters for ACO. For example in the parameter set labeled DS_ACO_1 the number of ants considered is 50, the tested value for the pheromone evaporation rate is 0.4 and value.
considered for the pheromone trail influence is 0.2. In order to determine the optimal value for these parameters different tests are made with Na values between 50 and 100, er values between 0.2 and 0.8 and α values between 0.2 and 0.8.

In Table III are introduced the AG parameters. For example in the parameter set labeled DS_AG_1 the considered maximum number of generations is 50, the number of individuals in the initial generation is 50, the crossover probability is 0.4 and the mutation probability is 0.04. In order to determine the optimal value for these parameters different tests are made with Gmax values between 50 and 100, n0 values between 50 and 100, pc values between 0.2 and 0.8 and pm values between 0.04 and 0.18.

Table V presents the experimental results acquired after testing the 3 proposed algorithms for the set of 15 classical benchmarks from the scientific literature. For each benchmark the algorithm runs with all 15 specific input parameters and for each combination a set of 50 tests is made.

The experiments have been made on a computer that has the following hardware configuration: AMD FX™-6100 Six-Core Processor, 3.30 GHz CPU, and the software solution was designed in C++ using QTCreator version 2.6.2.

The results of the simulations are synthetized in Table V that also includes the lower and upper bound and also the optimal value for each benchmark, as they are known in the scientific literature [29]. The next columns present the values obtained by running the evolutionary algorithms (PSO, ACO and AG) proposed in this paper.

The tested evolutionary algorithms have not obtain the optimal value but at least one of them fits successfully in the area of variation between the upper and lower bound in 100% of cases.

PSO obtains values that fit into the interval delimited by the lower bound and upper bound in 86.66% of the cases, and manages to obtain the best value of the tested algorithms in 80% of the cases.

The results obtained by ACO reveals the fact that it manages to fit between the upper bound and lower bound values in 66.66% of the cases, and in 20% of the cases obtains the best value of the tested algorithms.

AG achieves values that fit into the interval delimited by the lower bound and upper bound in 73.33% of the cases, and succeeds in reaching the best value of the tested algorithms in 26.66% of the cases.
As it was mentioned before, the specific parameters have a high importance in the algorithms behavior through the experimental tests. The Table VI presents the parameters set label for PSO, ACO and AG that correspond to the best obtained solution. For instance, for ABZ5 PSO obtains the best results (1350) with the parameters set DS_PSO_12, ACO reaches the result 1410 by running with DS_ACO_12, and AG obtains 1340 for the parameter set DS_AG_4.

<table>
<thead>
<tr>
<th>Benchmark Label</th>
<th>PSO Parameters set label</th>
<th>ACO Parameters set label</th>
<th>AG Parameters set label</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABZ5</td>
<td>DS_PSO_12</td>
<td>DS_ACO_10</td>
<td>DS_AG_4</td>
</tr>
<tr>
<td>ABZ6</td>
<td>DS_PSO_13</td>
<td>DS_ACO_4</td>
<td>DS_AG_14</td>
</tr>
<tr>
<td>FT6/MT06</td>
<td>DS_PSO_4</td>
<td>DS_ACO_11</td>
<td>DS_AG_15</td>
</tr>
<tr>
<td>FT10/MT10</td>
<td>DS_PSO_13</td>
<td>DS_ACO_11</td>
<td>DS_AG_4</td>
</tr>
<tr>
<td>LA01</td>
<td>DS_PSO_12</td>
<td>DS_ACO_10</td>
<td>DS_AG_4</td>
</tr>
<tr>
<td>LA02</td>
<td>DS_PSO_4</td>
<td>DS_ACO_11</td>
<td>DS_AG_4</td>
</tr>
<tr>
<td>LA03</td>
<td>DS_PSO_12</td>
<td>DS_ACO_10</td>
<td>DS_AG_15</td>
</tr>
<tr>
<td>LA04</td>
<td>DS_PSO_12</td>
<td>DS_ACO_11</td>
<td>DS_AG_15</td>
</tr>
<tr>
<td>LA19</td>
<td>DS_PSO_12</td>
<td>DS_ACO_12</td>
<td>DS_AG_14</td>
</tr>
<tr>
<td>LA20</td>
<td>DS_PSO_4</td>
<td>DS_ACO_11</td>
<td>DS_AG_4</td>
</tr>
<tr>
<td>ORB1</td>
<td>DS_PSO_13</td>
<td>DS_ACO_11</td>
<td>DS_AG_14</td>
</tr>
<tr>
<td>ORB2</td>
<td>DS_PSO_13</td>
<td>DS_ACO_11</td>
<td>DS_AG_4</td>
</tr>
<tr>
<td>ORB3</td>
<td>DS_PSO_12</td>
<td>DS_ACO_12</td>
<td>DS_AG_15</td>
</tr>
<tr>
<td>ORB4</td>
<td>DS_PSO_12</td>
<td>DS_ACO_11</td>
<td>DS_AG_15</td>
</tr>
<tr>
<td>ORB5</td>
<td>DS_PSO_13</td>
<td>DS_ACO_10</td>
<td>DS_AG_14</td>
</tr>
</tbody>
</table>

The three images highlights once again that evolutionary algorithms are suitable for the proposed area and managed to obtain great results in a considerable number of cases and even get very close to the recognized optimal value.

CONCLUSIONS

Due to his complexity, Job Shop Scheduling Problem is included into the NP-complex category by the researcher’s community and various sets of methods and algorithms were tested in order to find the best approach to solve it.

This paper presents an overview on the implementation of three algorithms (Genetic Algorithms, Particle Swarm Optimization and Ant Colony Optimization) for JSSP. Those algorithms are tested using a set of 15 classical benchmarks from the scientific literature. The results highlighted the fact that these methods are suitable for the proposed problem. This affirmation is sustained by the fact that PSO solutions manages to fit between the lower and upper bound values in 86.66% of cases, ACO in 66.66% of cases and AG in 73.33% of cases.

Simultaneously a significant disadvantage relates to the fact that the optimal value is never reached for the tested benchmarks.

Of the tested algorithms the best results are far away obtained by PSO (in 80% of the cases), while AG and ACO obtains similar but weaker results.

In future the research work will be focused in finding an hybrid algorithm based on the method that proved to be more appropriate for JSSP (PSO) and others evolutionary algorithms. The goal is to increase the algorithm performance and to reach as close as possible to the known optimal. Another future work direction is related to including a larger set of benchmarks to the tests.

REFERENCES


[27] S. Lawrence, “Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (Supplement)”, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pennsylvania, 1984


