Abstract — The Flow Shop Scheduling Problem (FSSP) is notoriously NP-hard combinatorial optimization problem. The goal is to find a schedule that minimizes the makespan. This paper proposes an adaptation of a new approach called Golden Ball Algorithm (GBA). The proposed algorithm has been never tested with FSSP; it’s based on soccer concept to obtain the optimal solution. Numerical results are presented for 22 instances of OR-Library. The computational results indicate that this approach is practical for small OR-Library instances.

Keywords — Combinatorial Optimization, Flow Shop Scheduling Problem, Golden Ball Metaheuristic, Makespan Time, Metaheuristics.

I. Introduction

The flow shop scheduling problem (FSSP) [1] is to schedule a set of n jobs on a set of m machines. The both parameters n and m are given. Each job consists of a chain of operations. All machines should process the jobs in the same order of jobs.

The objective is to schedule jobs in such a way as to minimize the maximum of the completion time of all the jobs (makespan). In the FSSP some constraints must be satisfied such as:

- All jobs are independent and ready for processing at time zero
- No n jobs may be processed at the same time on the same machine
- No preemption of a given job
- The precedence relations have to be respected
- The makespan of all jobs should be minimized

The FSSP is one of the most difficult combinatorial optimization problem and it belongs to NP-hard problems [2]. Finding a good solutions to the FSSP is of great interest to the industrial sector.

Several evolutionary methods are used to solve the flow shop scheduling problem such as: genetic algorithms [3]-[4], ant colony optimization [5], partial swarm optimization [6]-[7], Tabu search method [8]-[3] etc.

In this paper we present an adaptation of the Golden Ball algorithm (GBA) to the flow shop scheduling problem (FSSP). This algorithm is based on the soccer concepts. The objective of this approach is to produce satisfactory results more near optimal and in less time.

The rest of this paper is organized as follows: In section II, flow shop scheduling problem formulation. In section III, the golden ball metaheuristic. In section IV, the golden ball adaptation. In section V, experimental results on 22 OR-Library instances [9] and finally a conclusion.

II. Flow Shop Scheduling Problem Formulation

A flow shop scheduling problem (FSSP) can be defined by a set J of n jobs J= {J1,..., Jn} which have to be processed on a set M of m machines M= {M1,.....,Mm}. Each job consists of a series of operations Oik, where k indicates the machine Mk on which the operation must be processed and i defines the job to which the operation belongs. Each operation needs to be processed during an uninterrupted period of time on a given machine pik. Every machine processes the jobs in a fixed order of a given jobs. Each machine may process only one operation during the period of time. The objective of FSSP is to minimize the makespan Cmax of the whole process and to find the optimal schedule. Cmax is the completion time of all jobs.

We consider an example of FSSP with 3 machines and 4 jobs where:

M= {M1, M2, M3}, J= {J1, J2, J3, J4},

J1= {O11, O12, O13}
J2= {O21, O22, O23}
J3= {O31, O32, O33}
J4= {O41, O42, O43},

Each machine processes the jobs in this order: {J1, J2, J4, J3}

TABLE I shows the processing time of all operations:

<table>
<thead>
<tr>
<th>JOBS</th>
<th>OPERATION TIMES</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1(O11, O12, O13)</td>
<td>6 2 5</td>
</tr>
<tr>
<td>J2(O21, O22, O23)</td>
<td>2 3 2</td>
</tr>
<tr>
<td>J3(O31, O32, O33)</td>
<td>4 7 3</td>
</tr>
<tr>
<td>J4(O41, O42, O43)</td>
<td>1 3 1</td>
</tr>
</tbody>
</table>

The makespan is calculated using the Gantt chart representation (Fig.1):

In this example the best schedule obtained is:

{J1, J4, J3, J2}

with a minimal makespan Cmax=23.

In this section we describe the operation of a new metaheuristic recently proposed called Golden Ball (GB). This method is based on several concepts of soccer to find the optimal solution. It was proposed firstly in 2013 by E.Osaba et al. [10]. The recent version of GB was published in 2014 by the same authors [11]. This technique divides the initial solutions into groups. Each group represents a team. Each team works independently and competes with other teams to get the best solution.

Golden Ball algorithm is based on 4 main phases (Fig.2): Initialization phase, Training phase, Competition phase and Transfer phase.
III. THE GOLDEN BALL ADAPTATION

In this section we present the equivalence of main terms used to solve the FSSP with the GBA.

Player: Schedule
Team: Group of schedules
NT: Number of groups of schedules
NP: Number of schedules per group
Quality: Completion time of schedule (Cmax)
Strength value: Average completion time of each group, it is equal to the sum of all Cmax divided by NP
Coach: Training function
Captain: Best schedule of the group

There are two kinds of training function: Conventional trainings and Custom trainings.

We use the following methods as conventional training functions:

- 2-opt [12]-[13]: a local search operator highly used by several researches to solve TSP. Its goal is to improve the path by replacing in each step of 2-opt two edges by two other edges.

- Insertion method [14]: choose and extract a random operation, insert it in position k between i and j in a way that the whole schedule is improved.

- Swapping mechanism [15]:
  - First possible swap: select and swap two random operations.
  - Second possible swap: reverse the operations order between two random positions.
  - Third possible swap: divide the path into three parts. Copy the last part into the first part of the new schedule. Copy the second part into the second part of the new schedule and copy the inverse of the first part into the last part of the new schedule.

As a custom training function the proposed adaptation uses the Ordered Crossover (OX) [16].

Step1: select randomly two positions pos1 and pos2 to select the substring to be copied.

Step2: copy the substring from the parent into the corresponding position of the child schedule.

Step3: starting with the pos2, select all operations which are not already in the substring from the second parent.

Step4: place these operations into the positions of the child schedule from the left to the right according to the order of the sequence.

In the competition phase we order the schedules of each group according to their makespan in ascending order.

In the transfer phase especially in the season transfer all groups of schedules must be sorted according to their average cost in ascending order.

<table>
<thead>
<tr>
<th>GB ALGORITHM STEPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Determine the value of NT and NP.</td>
</tr>
<tr>
<td>2. Generate NT*NP random schedules</td>
</tr>
<tr>
<td>3. To each group assign randomly NP schedules and a training function.</td>
</tr>
<tr>
<td>4. For each group find its best schedule and calculate its average completion time.</td>
</tr>
<tr>
<td>5. Start of the season and points initialized to 0.</td>
</tr>
<tr>
<td>6. Start the training session for each group.</td>
</tr>
<tr>
<td>7. If the session is finished, sort schedules of each group according to their Cmax in ascending order. Else, start the transfer phase if it is necessary and go to step 6.</td>
</tr>
<tr>
<td>8. Compare each schedule of the group with another existing in other group chosen randomly. The group who has the better schedule receives 3 points. If the two schedules are equal the both groups receive 1 point.</td>
</tr>
<tr>
<td>9. If the season is not finished, go to step 4. Else,</td>
</tr>
<tr>
<td>If the optimal solution is found, stop the program.</td>
</tr>
<tr>
<td>Else,</td>
</tr>
<tr>
<td>• sort all groups according to their points and average completion time in ascending order.</td>
</tr>
<tr>
<td>• exchange schedules and training functions between groups.</td>
</tr>
<tr>
<td>• go to step 4.</td>
</tr>
</tbody>
</table>

IV. NUMERICAL RESULTS

The program is tested on different instances of OR-library. The GB algorithm was implemented in C language and compiled using Microsoft Visual Studio 2008, the program code was executed in computer with Genuine Intel( R ) 575 @ 2.00 GHz 2.00 GHz RAM 2,00 Go.

The program uses four training functions and two parameters: NT (number of groups), NP (number of schedules per group). NP is assumed as constant. The effect of NT parameter is evaluated in the test problems with 6 various values.

The parameters values in the TABLE II produce better results during the algorithm run. The results obtained for the different values of NT parameter are shown in the TABLE III. The application is run five times for each instance. The program stops after 60s.

<table>
<thead>
<tr>
<th>TABLE II</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARAMETERS VALUES</td>
</tr>
<tr>
<td>NT</td>
</tr>
<tr>
<td>NP</td>
</tr>
<tr>
<td>Maximum execution time of the program</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE III</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESULTS OBTAINED FOR EACH INSTANCE</td>
</tr>
<tr>
<td>NT</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
</tbody>
</table>
Special Issue on Artificial Intelligence Underpinning

The table above (TABLE IV) represents the following information:

- **BKS**: Best known Solution
- **Best**: Best schedule
- **NBest**: The number of times the algorithm reaches the best schedule
- **Worst**: The worst schedule
- **Average**: The average cost
- **%Error**: The percent error is calculated as follows:

\[
\%\text{Error} = \frac{\text{Average} - \text{BKS}}{\text{BKS}} \times 100
\]

The application is run ten times for each test instance. The program stops when the best solution is reached. The maximum execution time of the application is 500s.

The following table (Table V) compares the proposed approach (GBA) with other existed algorithms in the literature of flow shop scheduling problem such as Palmer [17], CDS [18], and NEH [19].

As the results show, the GBA is an effective algorithm for the flow shop scheduling problem.

### V. Conclusion

This paper presents a new approach called Golden Ball algorithm (GBA) for the flow shop scheduling problem (FSSP). This proposed technique is based on soccer concept to find the optimal schedule with a minimal makespan. GBA is able to very quickly find the optimal schedule for the small flow shop schedule problem. For the other OR-Library instances, the algorithm produces results near optimal. The future work may be to increase the performance and the quality of the proposed adaptation for the large instances.

### References


Ms. Fatima Sayoti (15-02-1988) received a master’s degree of software quality from University of Chouaib Doukkali, El Jadida, Morocco in 2011. She is currently pursuing his Ph.D. degree (Computer Science) at the University of Chouaib Doukkali, Faculty of Sciences, El Jadida, Morocco. She is also a temporary teacher at Polydisciplinary Faculty, El Jadida, Morocco.