A Hybrid Metaheuristic Algorithm for Flexible Job Shop Scheduling "PSO&TS"

Asen Tochev

Institute of Information and Communication Technologies – BAS, Sofia, Bulgaria

Kick-off Meeting for CEEPUS Network: CIII-BG-1103-01-1617
05-09 October 2016, Technical University-Sofia, Bulgaria

PSO&TS

The flexible job shop scheduling problem (FJSSP):

- An extension of classical JSSP taking into account the production flexibility;
- Unlike the classical JSSP, where each operation is processed on a predefined machine, each operation Oi_kij in the FJSSP can be processed on one machine M_l out of several machines $M_l \in Mij$, $Mij \subseteq M$.
- The objective is to **minimize the makespan** (the time for processing all the jobs).

PSO&TS

Flexible job shop scheduling problem (FJSSP) can be decomposed naturally into two tasks:

1) **object routing (appointment)** at which each transaction is assigned to a corresponding machine chosen among the set of machines, on which its processing is possible;

2) **subtask scheduling (sequence)** where you have to calculate the start times of the appointed operations of all machines to obtain acceptable schedule (which does not violate the technological limitations) and to optimize predefined criteria (objective functions).

PSO&TS

Properties of FJSSP:

- JSSP is NP hard;
- hence FJSSP is also NP hard;
- its optimal solution is difficult to be obtained even when the objective is to minimize the makespan (the time for processing all the jobs);

- For this reason, many studies are focused on the development of heuristic procedures for this problem.

- Well known among them are the Particle Swarm Optimization (PSO) algorithm (Ge et al. 2005, Zhang et al. 2009) and the metaheuristic Tabu Search (TS) (Saidi-Mehrabad & Fattahi 2007, Li et al. 2010, Jia & Hu 2014).

Subprocedure PSO

One powerful heuristic is the Particle Swarm Optimization (PSO). It has been first developed by Eberhart and Kennedy.

- The basic idea in PSO is to imitate the intelligent swarming behavior, observed in flocks of birds, schools of fish, swarms of bees, etc;
- Each particle in PSO represents one solution and it makes steps from its current position to a new position.
- This movement is determined as a sum of three vectors: ***inertia***, ***competition*** **and** ***cooperation***.

Subprocedure PSO

- Suppose that the search space is *d*-dimensional, and then the *i*-th particle of the swarm can be represented by a *d*-dimensional vector, $xi = (xi1, xi2, \ldots, xij, \ldots, xid)$, $d = m * n$.
- The velocity of the particle can be represented by another vector, $vi = (vi1, vi2, \ldots, vij, \ldots, vid)$.
- The best solution achieved by the particle is denoted as $pi = (pi1, pi2, \ldots, pij, \ldots, pid)$, and the global best solution/particle is denoted by $g = (g1, g2, \ldots, gj, \ldots, gd)$.
- The particle updates its velocity and position by means of equation (1) and (2) as follows.
- $vi+1 = w*vi+b*Dr1*(pi-xi)+c*Dr2*(gi-xi)$ (1)
- $xi(k+1) = xi(k) + vi$ (2)

Subprocedure TS

Metaheuristic created by Fred Glover (1990):

- Perform a strategic search procedure;
- start from one solution;
- avoid cycles;
- store some characteristics of solutions or movements (steps in given directions) which are forbidden (tabu) for a certain number of iterations.
- In this manner is avoided the trap of local optimality.
- The list of forbidden characteristics or movements stored in the memory is called tabu list.
- Typically the stopping criteria correspond to the iterations limit and to the number of consecutive iterations without improving the best obtained solution.

Proposed phases of the new algorithm

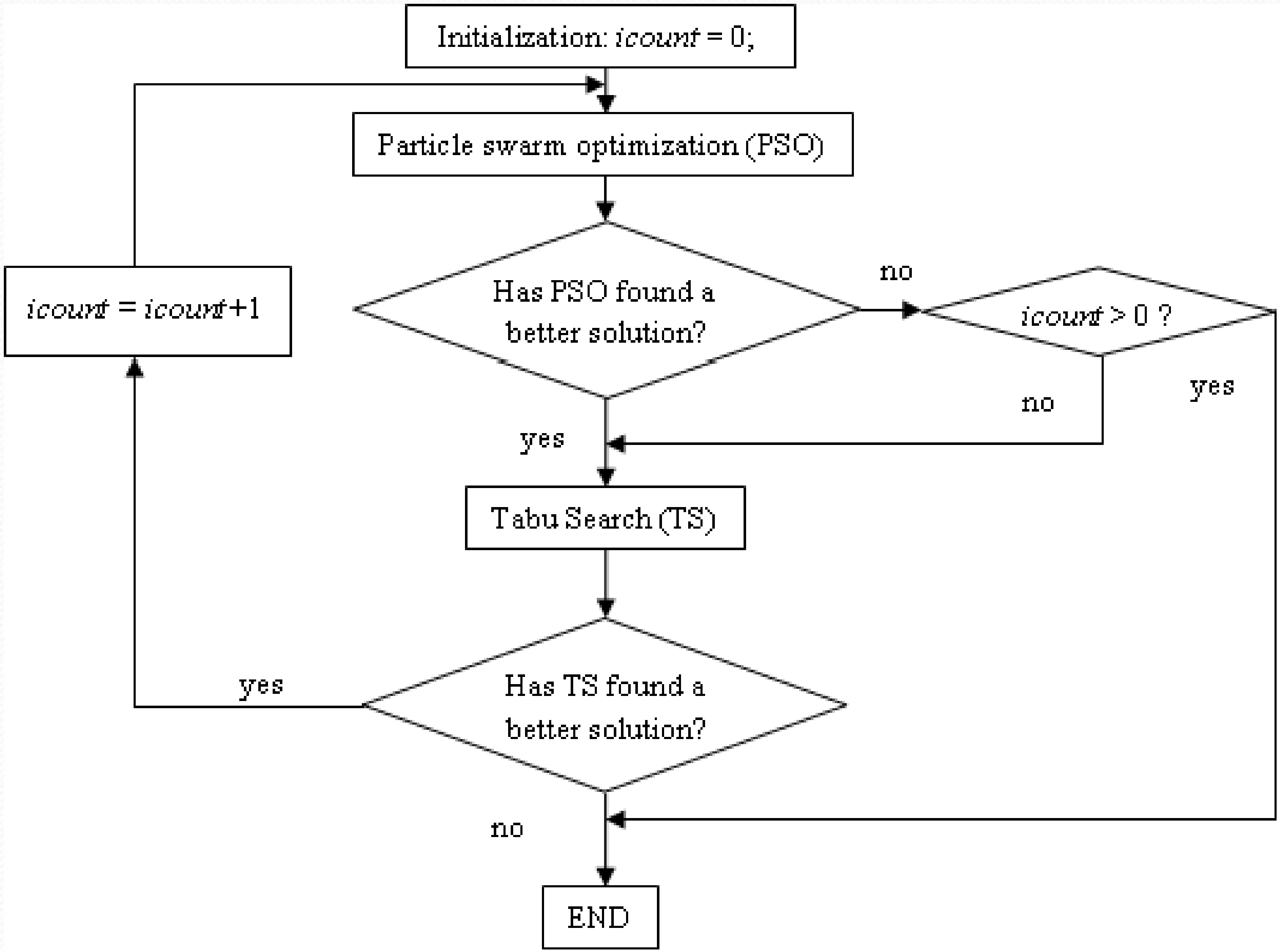
- **1)** Phase **Generation** of one initial solution like the procedure described in Hongwei Ge, Wenli Du, Feng Qian (2007).
- **2)** Phase **Initialization** of initial swarm in the PSO.
- **3)** Phase **PSO**.
- **4)** Phase **TS**.

Hybrid algorithm PSO & TS

Main idea:

- First a good solution to be found out by means of PSO heuristic.
- Then the "critical path" (the sequence of operations on the machine finishing last its processing) is assumed to become "tabu".
- After that a TS procedure is used, attempting to improve global_best generated by PSO.
- The phases 3) "PSO" and 4) "TS" are repeated iteratively until the stop condition is met.

Flow chart of PSO&TS algorithm



Illustrative example

In the paper it is solved an example of FJSSP of Fattahi et al. (2007) called M2J2O4, where:

- M means machines;
- J – jobs; .
- O – operations. .

Illustrative example

p_{ijh}		M_1	M_2
J_1	O_{11}	25	37
	O_{12}	32	24
J_2	O_{21}	45	65
	O_{22}	21	65

Illustrative example

1) $S_{km} = 0$ for every $O_{km} \in A$, Then

$S_{11} = 0$ for $O_{11} \in A$

$S_{21} = 0$ for $O_{21} \in A$

$t(A) = \min(s_{km} + p_{km}) = \min(25,65) = 25$

$M = \{0\}$, $(k,m) \in A$

$O_{11} \in A$. $SO_{11} = 25 = t(A)$. It follows that SO_{11} is not smaller of t(A).

$O_{21} \in A$. $SO_{21} = 25 = t(A)$. It follows that SO_{21} is not smaller of t(A).

Then $G = \{0\}$. m^*=M2

O_{21} runs on machine 2. Then $G = \{O_{21}\}$, $(k,m^*) = (k,2) = O_{21}$

Earlest time of $O_{21} = 65$.

$k^* = J_2$

$A = \{O_{11}\}$

$A = \{O_{11}, O_{22}\}$

$R = \{O_{21}\}$

With this ends the first iteration. Makespan = 130.

Illustrative example

On second iteration:

$R = \{O_{21}, O_{22}\}$

On third iteration:

$R = \{O_{21}, O_{22}, O_{11}\}$

On forth iteration:

$R = \{O_{21}, O_{22}, O_{11}, O_{12}\}$ is the initializing schedule for PSO.

2) Initial schedule $R = \{O_{21}, O_{22}, O_{11}, O_{12}\}$.

3) PSO:

The schedule $R = \{O_{21}, O_{22}, O_{11}, O_{12}\}$ is not improved.

4) TS:

1) O_{11}, O_{12} are run on M_1. O_{21}, O_{22} – on M_2. Makespan = 130.

O_{21}, O_{11}, O_{12} are run on M_1. O_{22} – on M_2. Makespan = 102.

This is better than 130.

O_{11}, O_{12}, O_{22} are run on M_1. O_{21} – on M_2. Makespan = 86.

O_{12} is run on M_1. O_{11}, O_{21}, O_{22} – on M_2. Makespan = 167.

O_{11} is run on M_1. O_{21}, O_{12}, O_{22} – on M_2. Makespan = 154.

Illustrative example

2) O_{11},O_{12}, O_{22} are run on M_1. O_{21} – on M_2.
Makespan = 86. Minimum.

O_{11},O_{12},O_{21},O_{22} are run on M_1. M_2 - deadtime.
Makespan = 187.

O_{12},O_{22} are run on M_1. O_{11}, O_{21} – on M_2. Makespan = 167.

3) O_{11},O_{12} are run on M_1. O_{21}, O_{22} – on M_2.
Makespan = 130. Minimum.

O_{22} is run on M_1. O_{11}, O_{21}, O_{12} – on M_2. Makespan = 126.

O_{11},O_{21},O_{22} are run on M_1. O_{21} – on M_2. Makespan = 91.

O_{11},O_{12},O_{22} are run on M_1. O_{21} – on M_2. Makespan = 130.

O_{21},O_{22} are run on M_1. O_{11}, O_{12} – on M_2. Makespan = 66.
Minimum. Solution. (The optimal solution is obtained.)

Conclusion

- The received results are encouraging.
- The PSO&TS algorithm has good performance and could be applied to solve large size Job Shop Scheduling problems.
- As a future step of this study PSO&TS will be tested on a set of benchmark examples.
- An important direction for further research is the multi-objective case of FJSSP to be considered. Possible criteria are: 1) Maximal completion time -Makespan, 2) Maximal machine idle time, 3) Total workload of machines, 4) Maximal tardiness of jobs

A.Tochev

THANK YOU!