KKU Engineering Journal

https://www.tci-thaijo.org/index.php/easr/index

Published by the Faculty of Engineering, Khon Kaen University, Thailand

Hybrid particle swarm optimization with a Cauchy distribution for solving a reentrant flexible flow shop problem with a blocking constraint

Chatnugrob Sangsawang and Kanchana Sethanan*

Research Unit on System Modeling for Industry, Department of Industrial Engineering, Faculty of Engineering, Khon Kaen University, Khon Kaen, 40002, Thailand

Received June 2015
Accepted September 2015

Abstract

This paper addresses a problem of a two-stage flexible flow shop with reentrant and blocking constraints in Hard Disk Drive Manufacturing. The problem can be formulated as a deterministic FFS|stage = 2, rcrc, block|Cmax problem. In this study, adaptive hybrid particle swarm optimization with a Cauchy distribution (HPSO) was developed. The objective of this research is to find the sequences that minimize the makespan. To demonstrate their performance, computational experiments were performed on a number of test problems and the results are reported. Experimental results showed that the proposed algorithms gave better solutions than the classical particle swarm optimization (PSO) for all test problems. Additionally, the relative improvement (RI) of the makespan solutions obtained by the proposed algorithms with respect to those currently used was performed to measure the quality of the makespan solutions generated by the proposed algorithms. The RI results show that the HPSO algorithm can improve the makespan solution by an average of 14.78%.

Keywords: Hybrid particle swarm optimization, Reentrant flexible flow shop, Blocking constraint, Hard disk drive manufacturing

1. Introduction

This paper presents a case study examining the activities of company which is involved in the production of hard drive disks (HDDs). Since the manufacture of HDDs requires a process which encompasses a number of uncertain elements, it is essential to incorporate a testing component to the process to ensure that every finished HDD meets specific functional requirements and standards. These standards must be measured, along with an evaluation of the product's reliability, before the HDD can be accepted for packaging and distribution. These stringent demands necessitate a procedure whereby every single completed HDD must be assessed using the testing machine. The testing machine can test a number of disks simultaneously, which have been loaded into the machine's slots by a robotic arm (see Figure 1). When the disks are in position, the machine carries out the designated test in line with the specific type of disk under observation. Upon completion of the test, the robotic arm removes the disks from each slot and then loads the next disks into the now vacant testing machine. The drawback to this system, however, is that the robotic arm is not always able to load and unload disks immediately because it may already be occupied in loading or unloading other disks and this creates delays within the operation of the testing process. Clearly this operating limitation can result in inefficiencies in the HDD testing process, which in turn has an effect on the overall efficiency of the HDD production system. When delays or bottlenecks occur, productivity is decreased, so it is essential to find a means of scheduling the testing of disks in such a way as to minimize these delays by reducing waiting periods where the robotic arm is unavailable. From a theoretical standpoint, the problem can be viewed as a two-stage flexible flow shop. The first stage comprises the robotic arm and the second stage is the tester slots. The tester slots in the test machine can represent parallel machines, while the actions of the robotic arm in loading and unloading disks can represent a reentrant product flow.

The flexible flow shop scheduling with unidirectional product flow has been widely studied in the literature including from 2-stage flexible flow shop to multi-stage flexible flow shop. Two-stage hybrid flow shop scheduling problems which incorporated parallel machines at the second stage were examined by [1]. A multi-stage hybrid flow shop scheduling problem was considered by [2] who investigated a three-stage hybrid flow shop in the woodworking industry. Since an exact algorithm may take such a long computational time to derive a solution, a heuristic approach is proposed to solve such a problem, for instance, see [3] and [4]. However, there is not much research in the flexible flow shop scheduling problem with reentrant product flows. [5] suggested a real-time scheduling mechanism with decision tree to solve the reentrant hybrid flow shop in a thin film transistor-liquid crystal display (TFT-LCD). For large-size problems, metaheuristics and a hybrid approach are suggested to solve the reentrant flow shop with complexity

*Corresponding author. Tel.: +6681 553 6429
Email address: ksethanan@gmail.com
doi: 10.14456/kkuenj.2016.9

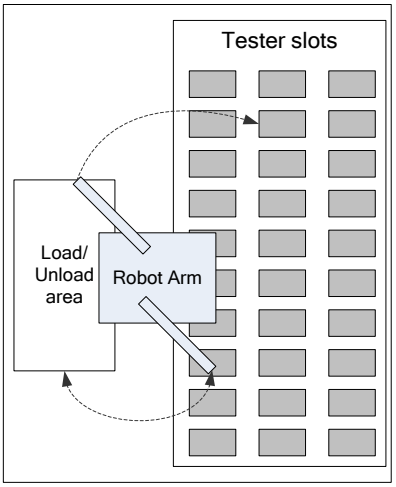

Figure 1 Operation of tester machine

and constraints. The metaheuristics are developed such as hybrid tabu search [6], and hybrid Genetic algorithm [7].

In addition to the consideration of the reentrant constraint, this research also considers the blocking constraint. Blocking scheduling denotes a process where jobs are processed in their entirety without interruption. This means that at each stage of the process, a job must wait at its current machine upon completion until the subsequent machine is available for the job to continue on its path. One such example can be seen in the steel industry, where the processing time of the operation is governed by the waiting times and the temperature [8]. Attempts to resolve these problems have typically involved the use of heuristics. However, metaheuristics can also be applied, [9] approached the large size restricted slowdown flow shop problem using a genetic algorithm. Meanwhile, [10] proposed a hybrid discrete differential evolution (HDDE) algorithm by combining the presented discrete version of the differential evolution and inserting a neighborhood based local search procedure. More recently, a modified genetic algorithm, ant colony optimization and a random search procedure were used to analyze and optimize the production planning of a bakery production line [11]. Although several studies address different constraints in the flexible flow shop problem in the literature, very few studies consider such realistic constraints simultaneously. Especially, there is no attempt to jointly consider complex recirculation (rcrc) and blocking (or no-wait; block) which are the realistic constraints included in the problem formulation of this paper. The hybridization of evolutionary algorithms has in recent years become a common means of addressing problems in industrial situations which encompass uncertainties [12-14]. The suggested method in this study is Hybrid Particle Swarm Optimization (HPSO) which combines the merits of classical PSO and local search. This approach can then have its efficiency assessed, while the threat of premature convergence is addressed by employing the Cauchy distribution. This allows exploitation and exploration relationships to be stimulated to the benefit of the PSO which relies on the nature of such relationships for its own performance. The organization of this paper is as follows. Section 2 introduces this particular case study with a description and definition of the problem. Section 3 provides the solution methodology involving the application of the Hybrid Particle Swarm Optimization (HPSO). Section 4 presents a numerical demonstration of the model by comparing a traditional PSO with the HPSO. The paper concludes with a summary in Section 5.

2. Problem description

Due to the test characteristics and complex machine operation, we can consider the test process to be a two-stage flexible flow shop environment. The first stage comprises one machine (the robot arm) and the second stage comprises a number of parallel machines (tester slots). The reentrant flow occurs in the first stage where the robot arm carries out two distinct actions: loading and then unloading the HDDs. The fact that the HDD enters the flow more than once allows the product flow to be designated as 'reentrant'. This is shown diagrammatically in Figure 2. From this figure, each job must pass 3 operations (O_{1a}, O_2, O_{1b}) with Operation O_{1a} and O_{1b} processed on the machine at Stage 1, while Operation O_2 is processed on one of the identical parallel machines in Stage2. According to [7], this is a problem of a type frequently observed in the production of HDDs. Once the operation 2 in Stage 2 is finished (i.e., HDD has finished the test), it must be processed in the machine at Stage 1 again (i.e., the HDD is unloaded) if the machine in Stage 1 is available (i.e., the robot arm is free). Otherwise, the HDDs must remain in the slot. Hence, a blocking constraint exists in this stage. Blocking often arises where production lines are operated in series but without the incorporation of buffer storage capacity. Without that buffer, blocking means that the unfinished product continues to occupy a machine until the path is clear for it to move to the next stage of the process [12]. Figure 3 demonstrates this situation, whereby in cases when Job 1 is completed by machine *m* it must then return to stage 1 before Job 2 is able to proceed to stage 2. Pinedo's 3 Field Notation can be used to express this type of problem. Flexible flow shop problems which blocking constraint and complex recirculation can be framed as an FFS|2-stage, rcrc, block|Cmax problem. This study aims to determine a job sequence which is able to minimize the makespan and thus improve the manufacturing efficiency for the HDD production.

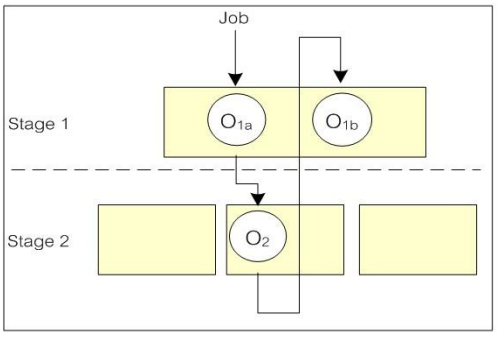

Figure 2 Reentrant flexible flow shop problem

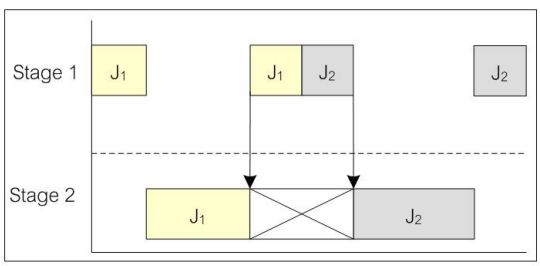

Figure 3 Blocking constraint

Dimension	1	2	3	4	5	6	7	8	9	10
$x_i(0)$	0.0	0.5	0.1	0.9	1.0	1.0	0.8	0.3	0.5	0.3
$v_i(0)$	3	-3	-3	1	2	1	0	3	-3	-2

Figure 4 Example of particle with job-based encoding routine

3. Hybrid particle swarm optimization

The metaheuristics approach can be applied extensively in many different scheduling problems, especially for complex and large-scale problems. A hybrid approach has been shown to obtain a better solution than the previous work applying the classical metaheuristics [15-16]. Recently, a novel evolutionary technique, named Particle Swarm Optimization (PSO), proposed by Kennedy and Eberhart [17], has gained much attention and wide application in a variety of fields. The PSO algorithm has a superior performance compared to available meta-heuristic algorithms [18].

Because there are limitations to the classical metaheuristics to solve complex problems efficiently, in this paper, the hybrid PSO developed to minimize the makespan for the two-stage reentrant flexible flow shop scheduling problem with blocking constraint. Details of the algorithms are presented below.

3.1 Encoding of particles

To find the solution by the PSO method, encoding of particles is one crucial issue to consider and affects the quality of the solution. Each particle represents the solution of one scheduling method. In this paper, a job-based encoding routine consists of the construction of the initial position ($x_i(0)$) and initial velocity ($v_i(0)$) of each particle in which its dimension (D) equals the number of jobs for each problem. An example of the problem with 10 jobs and *V*max equals 3 can be constructed the particle *k* as shown in Figure 4.

3.2 Fitness of particles

Once the particles are obtained, each particle is evaluated in order to find the objective of each problem which is the minimization of the makespan. The method to evaluate particles is the job-based decoding routine. From the example in Figure 4, the decoding method presented in Figure 5 is applied to determine the job orders of each dimension for the particle which is {1,3,8,10,2,9,7,4,5,6} (see Figure 6).

Each dimension represents the priority order of all operations (O_{1a}, O2, O_{1b}) of each job. The Gantt chart corresponding to the schedule can be generated as shown in Figure 7.

3.3 Update the position of the particle

The solution from using the PSO method can be determined by moving the particle in order to find the position with the best solution with experience of the particle. There are two experience positions used in the PSO: one is the global experience position of all particles (gbest), which memorizes the global best solution obtained from all positions (solutions) of all particles; the other is the individual experience position of each particle (hbest),

input: particle k, F(x)
begin
Generate Job all dimension for each particle;
Sorting all dimension of particle k by x_k ;
Create operation sequence for each job ;
for (job j ; first order to last order) **do**
In Stage 1, allocate job to earliest available time slot in Gantt chart;
In Stage 2, allocate job to machine which have earliest ready time ;
end;
output: evaluate value (*Cmax*)
end;

Figure 5 The procedure of the job-based decoding routine

which memorizes the local best solution acquired from the positions (solutions) the corresponding particle has been at. These two experience positions and the inertia weight of the previous velocities are used to determine the impact on the current velocity. Equation (1) will be used to change the velocity of the particle, while equation (2) will be used to update the position of the particle.

$$v_k(t+1) = w(t)v_k(t) + b_1r_1\left(h_{best_k} - x_k(t)\right) + b_2r_2\left(g_{best_k} - x_k(t)\right) \quad (1)$$

$$x_k(t+1) = x_k(t) + v_k(t+1) \quad (2)$$

where $v_k(t)$ is the k^{th} particle's flying velocity at the t^{th} iteration; b_1 and b_2 are positive constants, called the acceleration constants; and r_1, $r_2 \in [0,1]$ are uniform random numbers. Generally, the value of each component in *V* can be clamped to the range $[-V_{max}, +V_{max}]$ to control excessive roaming of particles outside the search area.

3.4 Combining with Cauchy distribution

After updating the position of particle according to Equation (2), we updated position by Cauchy distribution. The Cauchy distribution is applied for effective particle moving of particle which has a Gaussian-like peak with Lorentzian's wing for implying occasional long jumps among local samplings [19] according to Equations (3) and (4), respectively and the updating position in Equation (5).

$$u_k(t+1) = \frac{v_k(t+1)}{\sqrt{v_{k1}(t+1)^2 + v_{k2}(t+1)^2 .. + v_{kN}(t+1)^2}} \quad (3)$$

$$s_k(t+1) = u_k(t+1)\cdot\tan\left(\frac{\pi}{2}\cdot rand[0,1)\right) \quad (4)$$

$$x_k(t+1) = x_k(t) + s_k(t+1) \quad (5)$$

Dimension	1	2	3	4	5	6	7	8	9	10
$x_i(0)$	0.0	0.5	0.1	0.9	1.0	1.0	0.8	0.3	0.5	0.3
Job define	1	2	3	4	5	6	7	8	9	10

Dimension	3	4	5	7	9	10	1	6	2	8
$x_i(0)$	0.0	0.1	0.3	0.3	0.5	0.5	0.8	0.9	1.0	1.0
Job sequence	1	3	8	10	2	9	7	4	5	6

Figure 6 Illustration of particle with job-based decoding routine

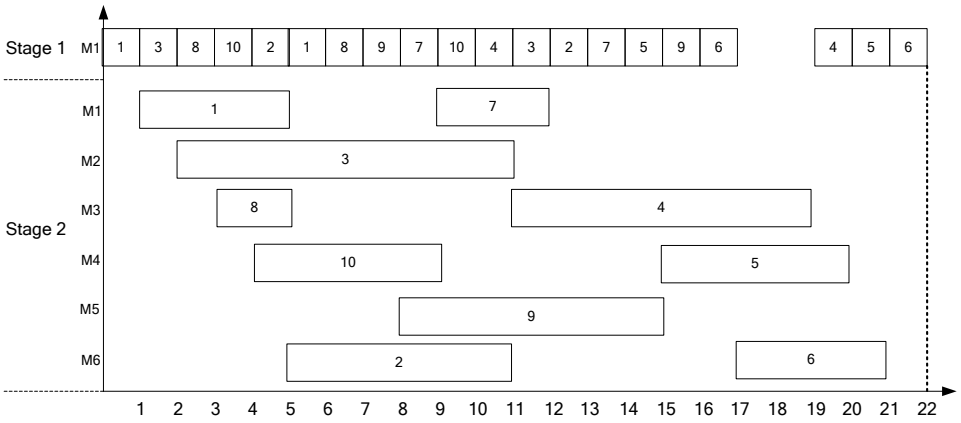

Figure 7 The gantt chart of example particle

where $u_k(t)$ and $s_k(t)$ are variables from updating position for each position k in generation t by Cauchy distribution in order to evaluate each particle and update $hbest_k$ and gbest values of the current solution.

3.5 Improve particle

Particle Improvement is a very important issue for the complex problem, especially the problem with reentrant flexible flow shop with blocking constraint, since in this problem, these two major constraints may cause the solution not to be optimal. In this PSO algorithm, particle improvement can be done by using the Relax-Blocking algorithm. To improve the decoded schedule after drawing the Gantt chart, a heuristic of local search, namely the Left-shift algorithm, was used for solving the reentrant flow shop (RFS) problem [20]. It will help to better minimize the makespan than the old Gantt chart. However, in this research, the relaxing blocking constraint is additionally added and is outlined in Figure 8. The concept of relaxing blocking is to release some operations to work independently rather than letting all three operations work as a group. In case of the operations O_{1a} and O_{1b} require processing simultaneously, the relaxing blocking constraint gives the priority to O_{1a} in order to input jobs to the system continuously (i.e., first stage machine) for processing.

3.6 Overall procedure of proposed HPSO

The overall procedure of HPSO is shown in Figure 9. Particle k is constructed by using the job-based encoding. Each particle consists of Velocity (v_k) and Position (x_k). The job-based decoding is applied to evaluate the particle. After that, position adjustment of the particle is used to find a better feasible solution for the problem based on its Global best experience and Local best of all other particles. The new velocity and position are updated based on equations (1) and (2), respectively, and also adjust the position by Cauchy distribution based on equations (3), (4), and (5). Then, the particle is improved using the Relax-Blocking algorithm to obtain a better solution, and the global best and local best are updated for determining the solution for the next iteration. To terminate the algorithm, the stopping criterion used in this study is the maximum number of iterations.

input: particle $X_k(t)$
begin
Transform all of offspring $C(t)$ by decoding routine;
Classify of the task in Stage 1;
Define set O which is the set of operation $O1a$ that can be move to left side without considering the $O1b$ operation;
while (set $O \neq 0$) **do**
Select the first job in set O and Check previous job in Stage 2;
Move operation O1a of selected job to insert at the position after O1b of the previous job in Stage 2;
Move operation O2 of selected job;
Delete the selected job from set O;
end;
Move left side for all operation O1b to feasible position;
output: improved schedule
end;

Figure 8 Procedure of the Relax-Blocking algorithm

```
input: RFFS problem data, PSO parameters (f(x), b1, b2, maxIter)
begin
  t→0;
  initialize (v_k, x_k) for each particle k by job-based encoding routine;
  evaluate x_k(t) by job-based decoding routine with block and keep the best solution;
   while (not termination condition) do
    for each particle x_k in swarm do
        update velocity v_k(t+1) by (1);
        update position x_k(t+1) by (2);
        calculate u_k(t+1) and s_k(t+1) by (3) and (4)
        update position x_k(t+1) using (5) ;
        improve particle k by relax-blocking routine;
        evaluate x_k(t+1) by job-based decoding routine;
      if f(x (t+1) ) < f(hbest_k) then
          update hbest_k = x_k(t+1);
      end
      gbest= argmin{f(hbest_k , f(gbest)};
       t→t+1;
    end
   output the best schedule gbest;
end;
```

Figure 9 Procedure of Hybrid Particle Swarm Optimization (HPSO)

4. Results and discussion

This section focuses on computational experience with the metaheuristic algorithms (i.e. PSO, and HPSO). The performance of the metaheuristic algorithms obtained by comparing their solutions (i.e., makespan) with respect to those of the current practice was investigated. In this case study, the company is focus on maximizing the utilization of the machine through schedule planning. The HDDs are scheduled to the tester machine based on the first come first serve (FCFS) policy. This currently involves the company using the Random Sequence with Earliest Available Time (RSEAT) technique. The RSEAT approach first allows the selection of the testing machine which has the highest available capacity, and then permits the selection of the slot with the earliest available test time.

Two sets of problems (set T1: 10 products and set T2: 20 products) were generated to evaluate the performance of the metaheuristic algorithms. Two types of data characteristics were generated for each set. The parameters for each data type, total number of machines at the second stage and processing times of products and deviations at the second stage, were randomly selected based on normalizing the real production time of the current practice. Since only one machine at the first stage is considered in this problem, the number machines at the second stage are varied and categorized into 2 levels: 6 machines and 12 machines. In the real problem, processing time of products at the second stage (i.e., testing operation) is very heterogeneous depending on product types, three levels of processing times and deviations are focused: (1) low processing time with low deviations (average processing time is 5 units and deviation is 2 units) (L), (2) medium processing time with medium deviations (average processing time is 15 units and deviation is 5 units) (M), and (3) high processing time with high deviations (average processing time is 25 units and deviation is 10 units) (H). For each combination, three test problems were generated.

Table 1 Parameters setting

Parameters	Value
Maximum Iteration : t_{max}	300
Number of particle : N	20
Maximum velocity : V_{max}	3
Inertia weight : w	0.5
Uniform random number : r_1 , r_2	0.2,0.3
Acceleration constants : b_1 , b_2	0.5

The proposed algorithms were run with MatLab on a 1.60 GHz PC, with 4 G-Byte of RAM, for testing and evaluation. To solve the problem using the proposed HPSO, the parameters used were set as shows in Table 1. To get the average performance of the HPSO algorithm, five runs on each test problem were performed and the makespan solutions were averaged. The makespan solutions were obtained for all combinations of sets and data types.

The computational results with the averages and the best solution obtained with the proposed algorithms are presented in Table 2. From this table we found that the HPSO yield better than the classical PSO for all test problems because the hybrid methods are satisfactorily embedded within reentrant and blocking constraints. In order to measure the quality of makespan solution generated by the proposed algorithms, the relative improvement (*RI*) of the makespan solutions obtained by the proposed algorithms with respect to those of the current practice were found. They were calculated by using Equation (6).

$$RI(\%) = (S_C - H_a) \times 100 / S_C \quad (6)$$

where H_a = the solution obtained from approach a (a = PSO or HPSO), S_c = the current practice solution.

The relative improvement results show that the PSO, and HPSO algorithms can improve the makespan solution by averages of 7.18%, and 14.78%, respectively (see Figure 10). We can notice that the HPSO gives the best solution.

5. Conclusion

This research addresses in this case study was that of the reentrant two-stage flexible flow shop with blocking constraint. It is a common problem in the manufacturing industry, especially in the production of electronics components. Previous studies addressing similar cases can be found which cover either the reentrant problem or the blocking aspect, but this study is the first to address these obstacles in combination. It is also the first study to employ hybrid algorithms, such as Hybrid Particle Swarm Optimization, in the search for an optimal solution which resolves both issues satisfactorily. The study aim to find the ideal means of minimizing the makespan in order to maximize efficiency. To accomplish this goal the study applied Job-base encoding and decoding along with an algorithm to address the blocking limitation. A Relax-Blocking algorithm was also employed to resolve the problems, while the Cauchy distribution is recommended in order to maximize the efficiency PSO. The results in this study were able to confirm that the application of these algorithms represents an effective means of solving this particular problem, and also suggest that the use of hybrid metaheuristics might represent a superior approach to

Table 2 Computational results

problem	Current practice	PSO		CPU Time*	HPSO		CPU Time*
		Avg.	Best		Avg.	Best	
1	28	23	22	2.12	20.4	20	1.54
2	26	22.4	21	1.58	20.6	20	1.39
3	25	23.2	22	2.14	20.4	20	2.03
4	25	22.6	22	3.32	20.4	20	3.14
5	25	23.4	23	3.21	20.6	20	2.45
6	26	23.6	22	4.09	20.2	20	3.11
7	54	51	50	2.59	42.6	42	2.47
8	52	49	48	3.02	40.4	40	2.44
9	55	49	48	3.11	42.8	42	2.51
10	48	45.2	44	5.32	40.4	40	5.17
11	46	43.4	42	4.58	40.4	40	4.22
12	48	44.6	44	5.21	40.2	40	4.19
13	75	71.6	70	3.07	60.2	60	2.33
14	77	70.2	69	2.49	62	60	2.36
15	72	70.8	70	3.19	62.4	62	3.14
16	68	66	65	4.33	60.4	60	3.38
17	69	63.6	63	5.34	60.2	60	3.12
18	68	63.8	63	5.22	61.2	60	4.06
19	142	132.8	132	2.46	122.2	122	1.47
20	140	131.2	130	3.03	121.4	120	2.08
21	135	129	128	2.51	120.8	120	2.19
22	136	128.4	127	6.1	120.4	120	5.36
23	132	127	126	5.32	121	120	4.42
24	134	105.6	127	5.29	122.4	122	4.51
25	127	113.6	112	2.37	100.6	100	1.49
26	122	113	112	3.19	102.8	102	2.55
27	125	112.6	111	3.02	100.4	100	2.31
28	116	114	113	6.11	101.8	102	5.43
29	118	109.6	109	6.32	102.4	102	5.22
30	115	108	106	5.54	100.4	100	4.11
31	226	212.4	211	3.26	206	204	2.53
32	223	212.8	212	4.12	205	204	3.22
33	227	214.6	214	4.03	202.6	202	3.26
34	218	207.2	206	6.11	202.4	202	4.47
35	220	207.4	207	6.53	200.8	200	3.54
36	217	207.8	207	6.17	201.6	200	5.51

*Computation time to obtain the best solution (minutes)

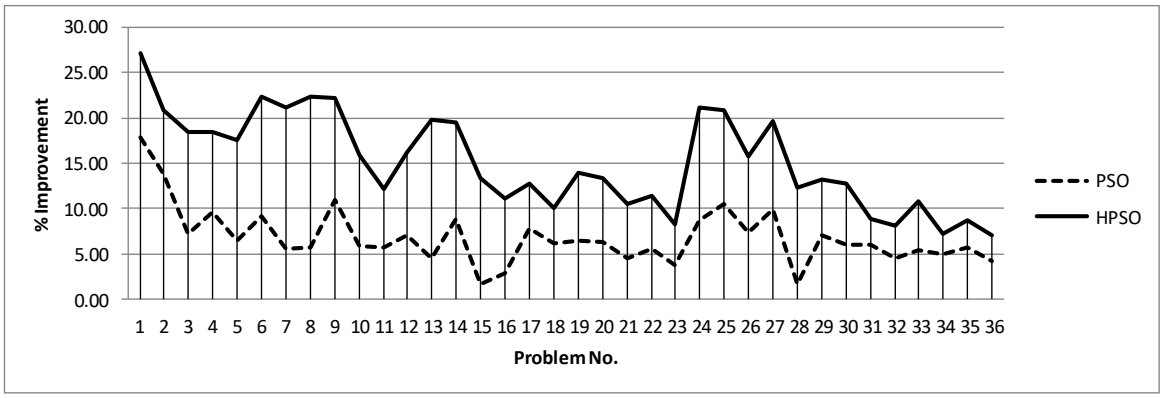

Figure 10 Percentage of relative improvement of each test problem

solving similar problems of this type in related fields since the performance shows improvement over traditional techniques. The RI results reveal that the use of Hybrid Particle Swarm Optimization algorithms improves the solution to the makespan problem by around 15%.

The results of this study might therefore prove useful in a number of extensions. In most industrial settings, scheduling problems arise in multiple stages, and it may thus be helpful to investigate FFS|stage=2,rcrc, block|Cmax in greater depth. Hybrid metaheuristics have shown their potential in this study, and further investigation in this field might add greater insights into the benefits and drawbacks of applying various algorithms. One final area which might be expanded in future work is that this particular study focused solely on minimizing the makespan, while real-world applications may be better served by algorithms which can simultaneously address a number of problems in seeking the optimal outcome for addressing multiple issues

6. Acknowledgements

This work was supported by the Thailand Research Fund through the Royal Golden Jubilee Ph.D. Program (Grant No. PHD/0123/ 2553) and also supported by the research unit on System modeling for Industry, Khon Kaen University, Thailand.

7. References

[1] Gupta JND, Tunc EA. Scheduling for a two-stage hybrid flowshop with parallel machines at the second stage. Int J Prod Res. 1991;29:1480-502.

[2] Riane F, Artiba A, Elmaghraby SE. A hybrid three-stage flowshop problem: efficient heuristics to minimize makespan. Eur J Oper Res. 1998;109:321-9.

[3] Lee J-S, Park S-H. Scheduling heuristics for a two-stage hybrid flowshop with nonidentical parallel machines. J Korean Inst Ind Eng. 1999;25:254-65.

[4] Wang I-L, Yang T, Chang Y-B. Scheduling two-stage hybrid flow shops with parallel batch, release time, and machine eligibility constraints. J Intell Manuf. 2012;23:2271-80

[5] Choi HS, Kim JS, Lee DH. Real-time scheduling for reentrant hybrid flow shops: a decision tree based mechanism and its application to a TFT-LCD line. Expert Syst Appl. 2011;38(4):3514-21.

[6] Chen JS, Pan JC-H, Wu CK. Hybrid tabu search for reentrant permutation flow shop scheduling problem. Expert Syst Appl. 2008;34(3):1924-30.

[7] Chamnanlor C, Sethanan K, Chien C-F, Gen M. Reentrant flow shop scheduling problem with time windows using hybrid genetic algorithm based on auto tuning strategy. Int J Prod Res. 2013;52(9):2612-29.

[8] Wagneur E, Sriskandarajah C. The two-machine permutation flow shop with state dependent processing times. Nav Res Logist Q. 1993;40:697-717.

[9] Caraffa V, Ianes S, Bagchi TP, Sriskandarajah C. Minimizing makespan in a blocking flowshop using genetic algorithms. Int J Prod Econ. 2001;70: 101-15.

[10] Wang L, Pan QK, Suganthan PN, Wang WH, Wang YM. A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems. Comput Oper Res. 2010;37(3):509-20.

[11] Hecker FT, Stanke M, Becker T, Hitzmann B. Application of a modified GA, ACO and a random search procedure to solve the production scheduling of a case study bakery. Expert Syst Appl. 2014;41(13): 5882-91.

[12] Luo H, Huang GQ, Zhang YF, Dai QY, Chen X. Two-stage hybrid batching flowshop scheduling with blocking and machine availability constraints using genetic algorithm. Robot Comput Integrated Manuf. 2009;25:962-71.

[13] Pongchairerks P, Kachitvichyanukul V. A particle swarm optimization algorithm on job-shop scheduling problems with multi-purpose machine. Asia Pac J Oper Res. 2009;26(2):161-84.

[14] Liao C-J, Tjandradjaja E, Chung TP. An approach using particle swarm optimization and bottleneck heuristic to solve hybrid flow shop scheduling problem. Appl Soft Comput. 2013;12:1755-64.

[15] Gao J, Sun L, Gen M. A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. Comput Oper Res. 2008;35: 2892-907.

[16] Gao J, Gen M, Sun L, Zhao X. A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems. Comput Ind Eng. 2007;53:149-62.

[17] Kennedy J, Eberhart R. Particle swarm optimization. Proceeding of IEEE International conference on Neural Network. Australia; 1995. p. 1942-8.

[18] Liu B, Wang L, Jin YH. An effective hybrid pso-based algorithm for flow shop scheduling with limited buffers. Comput Oper Res. 2008;35(9):2791-806.

[19] Wong TC, Ngan SC. A comparison of hybrid genetic algorithm and hybrid particle swarm optimization to minimize makespan for assembly job shop. Appl Soft Comput. 2013;13:1391-9.

[20] Li C, Liu Y, Zhou A, Kang L, Wang H. A fast particle swarm optimization algorithm with cauchy mutation and natural selection strategy. In: Kang L, Liu Y, Zeng S, editors. ISICA 2007: Advances in Computation and Intelligence; 2007 Sep 21-23; Wuhan, China. Berlin: Springer International Publishing; 2007. p. 334-43.