A hybrid cuckoo search metaheuristic algorithm for solving single machine total weighted tardiness scheduling problems with sequence dependent setup times

M.K. Marichelvam*

Department of Mechanical Engineering,
Kamaraj College of Engineering and Technology,
Virudhunagar 626001, Tamil Nadu, 626001, India
Fax: +91-4549-278172
E-mail: mkmarichelvamme@gmail.com
*Corresponding author

M. Geetha

Department of Mathematics,
Kamaraj College of Engineering and Technology,
Virudhunagar 626001, Tamil Nadu, 626001, India
E-mail: geethamkm1312@gmail.com

Abstract: In this paper, we present a hybrid algorithm based on cuckoo search algorithm to solve the single machine total weighted tardiness (SMTWT) scheduling problems with sequence dependent setup times which have been proved to be strongly NP-hard. Three different dispatching rules are incorporated with the initial random solutions of the cuckoo search (CS) algorithm to improve the solution quality. Computational results show that the proposed algorithm is very competitive to many metaheuristic algorithms in the literature.

Keywords: metaheuristics; cuckoo search; total weighted tardiness; NP-hard; scheduling.

Reference to this paper should be made as follows: Marichelvam, M.K. and Geetha, M. (2016) ‘A hybrid cuckoo search metaheuristic algorithm for solving single machine total weighted tardiness scheduling problems with sequence dependent setup times’, *Int. J. Computational Complexity and Intelligent Algorithms*, Vol. 1, No. 1, pp.23–34.

Biographical notes: M.K. Marichelvam is working as an Assistant Professor in the Department of Mechanical Engineering, Kamaraj College of Engineering and Technology, Virudhunagar, Tamilnadu, India. He received his BE in Mechanical Engineering from the Madurai Kamaraj University in 2000 and his ME in Industrial Engineering from the Thiagarajar College of Engineering, Madurai, India in 2002. His areas of interest are manufacturing scheduling, multi-objective optimisation, heuristics, and hybrid metaheuristics. Currently, he is pursuing his PhD at the Anna University, Chennai. He has published five papers in refereed journals.

M. Geetha is working as an Assistant Professor in the Department of Mathematics, Kamaraj College of Engineering and Technology, Virudhunagar, Tamilnadu, India. She received her BSc, MSc, and MPhil in Mathematics from the Madurai Kamaraj University, Madurai, India. Her area of interest is operations research. She has published a paper in an international journal.

1 Introduction

Scheduling plays an important role in manufacturing system to improve the performance. The limited resources are allocated over time to perform a collection of tasks in scheduling to optimise one or more objective function. The different types of scheduling environments can be found from Baker (1974). Many research works in scheduling have dealt with makespan as the objective function. But the due date related problems are much important in today's competitive environment (Dhingra and Chandna, 2010). This paper considers the SMTWT scheduling problems with sequence dependent setup times.

Kanet (1981) studied the single machine scheduling problem for minimising the sum of deviations from a common due date. He presented a polynomially bounded matching algorithm for the problem. Hoogeveen and Van de Velde (1991) proved the single machine scheduling problem with sequence dependent setup time was NP – hard. As enumeration methods cannot be used to obtain optimal solutions for these problems, researchers have applied many heuristics and metaheuristics like ant colony optimisation (ACO), genetic algorithm (GA), particle swarm optimisation (PSO), tabu search (TS), simulated annealing (SA), and memetic algorithm (MA) to solve these problems. Rubin and Ragatz (1995) proposed a GA to solve the single machine total tardiness scheduling problems with sequence dependent setup times. Lee and Kim (1995) presented a parallel GA to solve the earliness tardiness job scheduling problem with general penalty weights. Lee et al. (1997) proposed a constructive heuristics for the SMTWT scheduling problems with sequence dependent setup times. James (1997) developed a TS algorithm for solving the common due date early/tardy machine scheduling problem. Tan and Narasimhan (1997) proposed a SA algorithm for minimising tardiness on a single processor with sequence dependent setup times. Tan et al. (2000) also considered the single machine scheduling problem with sequence dependent setup times to minimise the total tardiness. They compared the performance of four different algorithms namely; branch-and-bound, genetic search, SA and random-start pair wise interchange algorithms. França et al. (2001) proposed a MA and also a GA for the total tardiness single machine scheduling problem. Gagne et al. (2002) developed an ACO algorithm for the single machine scheduling problem with sequence-dependent setup times. Feldmann and Biskup (2003) solved the single machine scheduling problems by evolutionary strategy, SA and threshold accepting approaches to minimise the weighted sum of earliness and tardiness.

Liao and Juan (2007) developed an ACO algorithm. They introduced a new parameter for initialising the pheromone trails. They also adjusted the timing of applying local search in their work. Valente and Alves (2008) have developed a beam search algorithm for the single machine scheduling problem with sequence dependent setup times. A discrete differential evolution algorithm was proposed by Tasgetiren et al. (2009). The performance of the algorithm was enhanced by applying different constructive heuristics. Anghinolfi and Paolucci (2009) developed a discrete PSO algorithm to solve the SMTWT scheduling problems with sequence dependent setup times. Ying et al. (2009) addressed an effective iterated greedy algorithm to solve single machine problems with sequence dependent setup times minimising the total weighted and un-weighted tardiness. A hybrid GA was proposed by Zhou et al. (2009) to minimise weighted tardiness in job shop scheduling problems. Valente and Schaller (2010) proposed an improved heuristics to minimise linear earliness and quadratic tardiness costs of single machine scheduling problems. Mahdavi Mazdeh et al. (2010) proposed three heuristic algorithms based on TS. Recently, Ahmadizar and Hosseini (2011) have proposed an ACO algorithm for the SMTWT scheduling problems with sequence dependent setup times.

Cuckoo search (CS) algorithm is a recently developed population-based metaheuristic algorithm developed by Yang and Deb (2009). Yang and Deb (2010) have also solved various optimisation problems using the CS algorithm. Noghrehabadi et al. (2011) applied a power series and CS optimisation algorithm to compute buckling of micro fixed- fixed beams. Valian et al. (2011) have developed an improved CS algorithm for training feed forward neural networks for two benchmark classification problem. They also suggested a proper strategy for tuning the CS parameters. Divya et al. (2011) proposed a cuckoo-based particle approach to achieve energy efficient wireless sensor network and multimodal objective functions. Recently, Marichelvam (2012) presented an improved hybrid cuckoo search (HCS) metaheuristic algorithm for permutation flow shop scheduling problems to minimise makespan. From the literature review one can easily understand that the application of the CS algorithm for the scheduling problems is very limited. Hence, in this paper an attempt is made to solve the SMTWT scheduling problems with sequence dependent setup times using a HCS algorithm.

The remainder of the paper is presented as follows. Problem definition is given in Section 2. The proposed HCS algorithm is described in Section 3. Computational results are given in Section 4. Conclusions and future research opportunities will be discussed in Section 5.

2 Problem definition

The problem of scheduling N jobs on a single machine with sequence dependent setup time is considered in this paper. The problem may be stated as follows. A set of n jobs $\{J_1, J_2, \ldots, J_n\}$ are available to be scheduled on a single machine. Each job has a processing time P_j with a due date d_j. The weight for each job is w_j and the setup time is s_{ij}. The setup time takes place when a job j immediately follows job i in the processing sequence. The tardiness of job j is T_j. The single machine total weighted tardiness problem with sequence dependent setup time is generally expressed as $1 / s_{ij} / \sum w_j T_j$. All these values are assumed to be non-negative integers. The mathematical model for the single machine total weighted tardiness scheduling problems with sequence dependent setup time is presented below. The model is due to Ahmadizar and Hosseini (2011). The following assumptions are made in this work.

1 the machine can process one job at a time

2 all jobs are available at time zero

3 each job is processed only once on the machine

4 no job pre-emption is allowed

5 job processing times and due dates are known in advance

6 due date is common for all jobs and is restrictive.

π is a processing sequence of jobs such that $\pi = \{\pi(0), \pi(1), \ldots, \pi(n)$, where π_m is the index of job m in the processing sequence and $\pi(0)$ is a dummy job and hence $\pi(0) = 0$. The completion time of the job scheduled at the k^{th} position may be calculated as follows.

$$C_{\pi(k)} = \sum_{l=1}^{k} \left(s_{\pi(l-1)\pi(l)} + p_{\pi(l)} \right). \tag{1}$$

The tardiness of job $\pi(k)$ in the k^{th} position of the sequence is:

$$T_{\pi(k)} = \max\left(0, C_{\pi(k)} - d_{\pi(k)}\right). \tag{2}$$

3 A novel HCS algorithm

For the past two decades researchers concentrated on many metaheuristic algorithms like GA, TS, SA, MA and PSO to solve the SMTWT scheduling problems with sequence dependent setup times. The CS algorithm is a recently developed metaheuristic algorithm. It has been developed by simulating the intelligent breeding behaviour of cuckoos. It is a population-based search algorithm like GA and PSO. The CS algorithm has been applied to solve different optimisation problems discussed in Introduction. But the applications of the CS algorithm to the scheduling problems are very limited. Hence in this paper, an attempt is made to apply the CS algorithm to solve the NP-hard type SMTWT scheduling problems with sequence dependent setup times. Moreover, in order to improve the solution quality, three different dispatching rules are incorporated with the initial solutions in the CS algorithm.

3.1 Proposed CS algorithm

CS is a new nature – inspired metaheuristic algorithm developed by Yang and Deb (2009). CS algorithm was inspired by the obligate brood parasitic behaviour of some cuckoo species in combination with the Lévy flight behaviour of some birds and fruit flies in nature. The following three idealised rules are considered for describing the CS algorithm.

1 each cuckoo lays one egg (solution) at a time, and dumps its egg in a randomly chosen nest

2 the best nests with high-quality eggs (solutions) will carry out to the next generation

3 the egg laid by a cuckoo can be discovered by the host bird with a probability P_a and a nest will then be built.

For minimisation problems the quality or fitness function value may be the reciprocal of the objective function. Each egg in a nest represents a solution and the cuckoo egg represents a new solution. The better solutions are replacing the worse solutions. The

Pseudo code of the CS algorithm was given by Yang and Deb (2009). The Pseudo code is depicted in Figure 1.

Figure 1 Pseudo code of the CS

```
Start
    Objective function f(x), x = (x_1, ..., x_d)^T
    Generate initial population of N host nests x_i (i = 1, 2,..., n)
    While (t < MaxGeneration) or (stop criterion)
        Get a cuckoo randomly by Lévy flights
            evaluate its quality/fitness F_i
        Choose a nest among N (say, e) randomly
        if (F_d > F_e),
            replace e by the new solution;
        End
        A fraction (P_a) of worse nests are abandoned and new ones are built;
        Keep the best solutions (or nests with quality solutions);
        Rank the solutions and find the current best
    end while
     Post process results and visualisation
End
```

When generating new solutions $x^{(t+1)}$ for a cuckoo *i* a Lévy flight is performed.

$$x_i^{t+1} = x_i^t + \alpha \otimes Lévy(\lambda) \tag{3}$$

where $\alpha > 0$ is the step size which should be related to the scales of the problem of interests. The value of α may be used as 1. The above equation is the stochastic equation for random walk. A random walk is a Markovian process in which the next iteration depends on the current iteration and the transition probability. The product $\otimes$ means entry-wise multiplication (or the Hadamard product). Entry-wise multiplication is not the usual inner production. In the entry-wise multiplication each term and entry are considered individually.

The random walk via Lévy flight is more efficient in exploring the search space. The random step length is drawn from the following Lévy distribution.

$$\text{Lévy} \sim u = t^{-\lambda}, 1 < \lambda \leq 3 \tag{4}$$

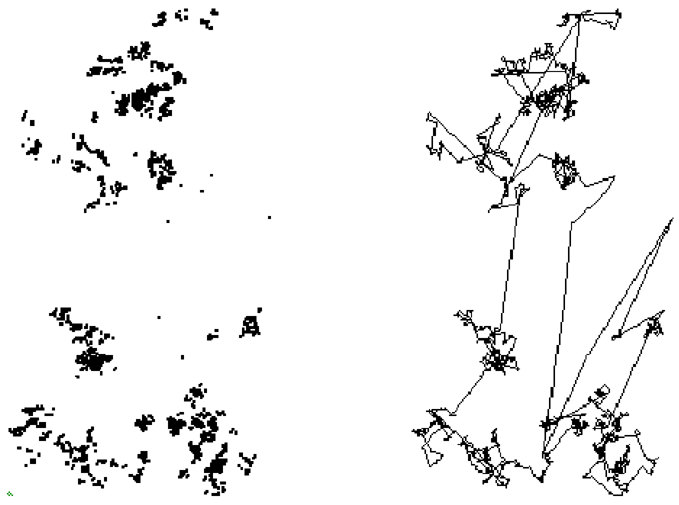
This has an infinite variance with an infinite mean. The Lévy flight provides a cuckoo with a random walk that has a power law step length distribution with a heavy tail. Some of the new solutions would be generated by Lévy walk around the best solution obtained so far, this will speed up the local search. However, a substantial fraction of the new solutions would be generated by far field randomisation and whose locations should be far enough from the current best solution, this will make sure the system will not be trapped in a local optimum (Yang and Deb, 2009).

When a solution is in infeasible region, this is managed by applying the constraints using penalty methods. So, when a constraint is violated, the objective will be penalised (with a very high value). In the CS algorithm, solutions are generated first, and then evaluate the objective and constraints so as to see if the solutions are feasible and improve or not.

3.1.1 Lévy flights

In nature, animals search for food in a random manner. In general, the foraging path of any animal is effectively a random walk because the next move is based on the current location and the transition probability to the next location. The direction of movement depends implicitly on a probability which can be modelled mathematically. The Lévy flights model involves a lot of small steps, interspersed with occasional very large excursions. Lévy flights are a random walk whose step length is drawn from Lévy distribution (Yang and Deb, 2011). The foraging path is very important as the stopping points of a Lévy flight are fractal (scale invariant is the main point here), and in complex ecosystems the distribution of food is also fractal (i.e., there are some large areas without food). To avoid spending too much time in such unproductive areas, animals need to develop search strategies that generate a fractal distribution of stopping points. Lévy flights have this property. The turning points and the trajectory of the Lévy flights are presented in Figure 2.

Figure 2 Turning points and trajectory of the Lévy flights



3.2 HCS algorithm for SMTWT scheduling problems

In most of the metaheuristics, the initial solutions are generated randomly. Vallada et al. (2008) prove that the dispatching rules would improve the solution quality for single machine scheduling problems to minimise tardiness. Hence, in this paper we attempt to incorporate three dispatching rules namely, the shortest processing time (SPT) rule, longest processing time (LPT) rule and earliest due date (EDD) rule with the initial random solutions to enhance the solution quality for the SMTWT scheduling problems with sequence dependent setup times. In the SPT rule, jobs are scheduled in the non-decreasing order of their processing times. But, in the LPT rule jobs are scheduled in the decreasing order of their processing times. According to EDD rule jobs are processed according to EDDs (Kim, 1993).

3.3 Solution representation

Each sequence of jobs is considered as a nest. The objective function is minimisation of tardiness. Three solutions are generated by using the dispatching rules and the remaining solutions are generated randomly.

For example, consider a 20 job problem. For this problem, 3 solutions are generated by using the dispatching rules and the remaining 17 solutions are generated randomly. A solution is represented graphically as shown in Figure 3.

Figure 3 Solution representation

5	4	3	14	17	16	11	6	18	2	19	7	15	20	8	10	12	9	1	13

4 Computational results

The HCS algorithm is coded in C++ and run on a PC with an Intel Core Duo 2.4 GHz CPU, 2 GB RAM, running Windows XP.

4.1 Design of experiments

To evaluate the performance of the proposed algorithm, design of experiments is carried out with different parameter settings. Table 1 gives the factor levels for the design of experiments to define the production systems. In this paper, we conduct $4 \times 3 \times 4 \times 4 \times 3 = 576$ experiments to compare the performance of the proposed HCS algorithm with other metaheuristics. Each problem instance is replicated 20 times with different initial solutions.

Table 1 Factor levels for the design of experiments

Sl. no.	*Factors*	*Levels*
1	Number of jobs	5, 10, 50 and 100
2	Processing time distribution	Uniform (1–10)
		Uniform (1–50)
		Uniform (1–100)
3	P_a	0.01, 0.1, 0.3 and 0.5
4	α	0.05, 0.1, 0.5 and 1.0
5	λ	1.5, 2.0 and 2.5

4.2 Comparison of different metaheuristics

The proposed HCS algorithm is compared with GA, SA, PSO, ACO, MA and CS algorithms. We use mean relative percentage deviation (MRPD) to measure the performance of the proposed algorithm. MRPD is calculated as follows.

$$MRPD = \sum_{b=1}^{R} \frac{\left(Solution_{best} - Solution_{ACO/GA/CS/HCS/PSO/MA/SA}\right)}{Solution_{best}} \times 100 / R \tag{5}$$

where R is the number of runs (index b). We consider the computational time also for comparing the results. The MRPD comparison of different metaheuristics for different size problems is given in Table 2. The mean computational time comparison of different algorithm for different size problem is presented in Table 3.

Table 2 MRPD comparison of different metaheuristics

Sl. no	*Number of jobs*	*MRPD*						
		ACO	*CS*	*GA*	*HCS*	*MA*	*PSO*	*SA*
1	5	1.54	0.76	1.25	0.0	2.06	0.91	1.93
2	10	1.73	0.80	1.32	0.0	2.18	1.02	2.04
3	50	1.56	0.94	1.28	0.0	2.19	1.13	2.08
4	100	1.54	0.66	1.30	0.0	2.32	0.82	2.10

Table 3 Computational time comparison of different metaheuristics

Sl. no	*Number of jobs*	*Computational time (seconds)*						
		ACO	*CS*	*GA*	*HCS*	*MA*	*PSO*	*SA*
1	5	5.23	4.84	5.16	4.78	5.17	4.91	5.44
2	10	7.85	6.53	7.69	6.41	7.49	7.12	7.88
3	50	31.40	27.29	30.84	26.86	30.38	28.87	31.96
4	100	44.58	37.46	43.79	36.75	44.35	40.42	45.38

From Table 2 it is concluded that the HCS algorithm provides better results than ACO, CS, GA, MA, PSO and SA algorithms.

4.3 Evaluation of benchmark problems

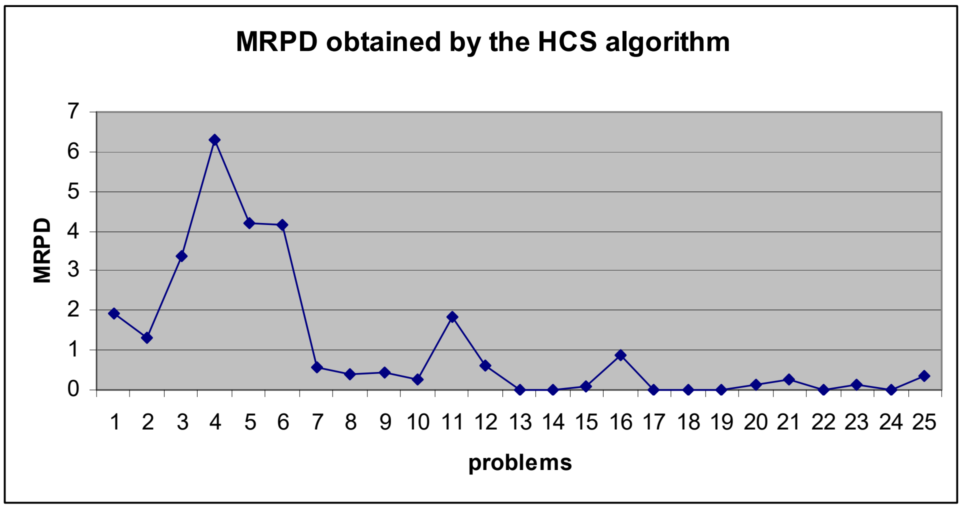
In order to evaluate the performance of the proposed algorithm, the benchmark problems studied by Cicirello and Smith (2005) are also considered in this paper. The benchmark problem consists of 120 problems each with 60 jobs. From the 120 problems 25 problems are randomly selected and the HCS algorithm is applied to the problems. Ahmadizar and Hosseini (2011) have also considered the similar problems. The results obtained by the improved CS algorithm are compared with other metaheuristics addressed in the literature (ACO, CS, GA, MA, PSO and SA).

From the result table it is noted that the HCS algorithm improves the solution for the benchmark problems significantly. The table also indicates that the HCS algorithm improves the solution for 80% of benchmark problems. For remaining 20% problems all the metaheuristics yield similar results. The MRPD obtained by the HCS algorithm for the benchmark problem is shown in Figure 4.

Table 4 Result comparison of the benchmark problems

Sl. no.	*Problem number*	*Best known solutions*	*Total weighted tardiness value*							*MRPD*
			ACO	*CS*	*GA*	*HCS*	*MA*	*PSO*	*SA*	
1	1	474	516	510	515	506	517	516	519	1.94
2	7	3,350	3,458	3,420	3,446	3,412	3,467	3,459	3,462	1.33
3	10	1,799	1,871	1,812	1,870	1,808	1,884	1,868	1,874	3.37
4	14	2,268	2,761	2,612	2,760	2,587	2,765	2,754	2,756	6.30
5	18	857	1,195	1,160	1,196	1,145	1,202	1,190	1,190	4.18
6	20	2,111	2,485	2,390	2,482	2,382	2,492	2,482	2,483	4.14
7	24	1,033	1,042	1,048	1,040	1,036	1,048	1,042	1,046	0.58
8	41	69,102	69,627	69,440	69,608	69,339	69,742	69,630	69,632	0.41
9	43	145,310	146,068	145,700	146,088	145,442	146,082	146,064	146,066	0.43
10	47	73,005	73,621	73,480	73,632	73,414	73,642	73,624	73,628	0.28
11	53	84,841	86,364	85,812	86,348	85,783	86,378	86,366	86,372	1.83
12	58	45,322	47,159	46,892	47,122	46,874	47,165	47,157	47,162	0.60
13	62	44,769	44,781	44,778	44,777	44,776	44,788	44,778	44,782	0.01
14	67	29,390	29,390	29,390	29,390	29,390	29,390	29,390	29,390	0.00
15	73	28,785	28,824	28,804	28,814	28,798	28,832	28,822	28,824	0.09
16	79	114,999	117,190	116,400	117,210	116,178	117,214	117,186	117,192	0.86
17	81	383,485	383,485	383,485	383,485	383,485	383,485	383,485	383,485	0.00
18	84	329,670	329,670	329,670	329,670	329,670	329,670	329,670	329,670	0.00
19	89	410,092	410,092	410,092	410,092	410,092	410,092	410,092	410,092	0.00
20	95	517,011	522,717	522,100	522,732	521,987	522,746	522,714	522,724	0.14
21	99	364,442	368,603	367,916	368,592	367,643	368,622	368,605	368,612	0.26
22	103	378,602	378,602	378,602	378,602	378,602	378,602	378,602	378,602	0.00
23	106	454,379	454,983	454,462	454,968	454,437	455,002	454,980	454,996	0.12
24	113	260,176	260,288	260,256	260,292	260,245	260,304	260,286	260,297	0.02
25	119	573,046	578,827	577,214	578,814	576,876	578,846	578,828	578,832	0.34

Figure 4 MRPD obtained by the HCS algorithm for the benchmark problems (see online version for colours)



5 Conclusions and future research

In this paper, a HCS algorithm is presented to solve the single machine total weighted tardiness scheduling problems with sequence dependent setup times. We incorporate three dispatching rules for initialisation so as to improve the solution quality. The proposed algorithm has been tested on benchmark problems taken form the literature and compared with other metaheuristics. Extensive computational experiments on a wide range of problem sets show that the proposed algorithm outperforms many other metaheuristics. Moreover, the proposed algorithm illustrates the significance of the dispatching rules. The proposed algorithm is applied to solve scheduling problems with single objectives only. It would be interesting to apply the CS algorithm for other type of scheduling problems. The algorithm may also be extended to solve multi-objective scheduling problems.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments and valuable suggestions that improved the quality of this paper.

References

Ahmadizar, F. and Hosseini, L. (2011) ‘A novel ant colony algorithm for the single – machine total weighted tardiness problem with sequence dependent setup times’, *International Journal of Computational Intelligence Systems*, Vol. 4, No. 4, pp.456–466.

Anghinolfi, D. and Paolucci, M. (2009) 'A new discrete particle swarm optimization approach for the single-machine total weighted tardiness scheduling problem with sequence-dependent setup times', *European Journal of Operational Research*, Vol. 193, No. 1, pp.73–85.

Baker, K.R. (1974) *Introduction to Sequencing and Scheduling*, 1st ed., John Wiley, New York.

Cicirello, V.A. and Smith, S.F. (2005) 'Enhancing stochastic search performance by value-based randomization of heuristics', *Journal of Heuristics*, Vol. 11, No. 1, pp.5–34.

Dhingra, A. and Chandna, P. (2010) 'A bi-criteria M-machine SDST flow shop scheduling using modified heuristic genetic algorithm', *International Journal Engineering Science & Technology*, Vol. 2, No. 5, pp.216–225.

Divya, M., Sundarambal, M. and Anand, L.N. (2011) 'Energy efficient computation of data fusion in wireless sensor networks using cuckoo based particle approach (CBPA)', *International Journal of Communications, Network and System Sciences*, Vol. 4, No. 4, pp.249–255.

Feldmann, M. and Biskup, D. (2003) 'Single-machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches', *Computers & Industrial Engineering*, Vol. 44, No. 2, pp.307–323.

França, P.M., Mendes, A. and Moscato, P. (2001) 'A memetic algorithm for the total tardiness single machine scheduling problem', *European Journal of Operational Research*, Vol. 132, No. 1, pp.224–242.

Hoogeveen, J.A. and Van de Velde, S.L. (1991) 'Scheduling around a small common due date', *European Journal of Operational Research*, Vol. 55, No. 2, pp.237–242.

James, R.J.W. (1997) 'Using tabu search to solve the common due date early/tardy machine scheduling problem', *Computers and Operations Research*, Vol. 24, No. 3, pp.199–208.

Kanet, J.J. (1981) 'Minimizing the average deviation of job completion times about a common due date', *Naval Research Logistics Quarterly*, Vol. 28, No. 4, pp.643–651.

Kim, Y.D. (1993) 'Heuristics for flow shop scheduling problems minimizing mean tardiness', *Journal of Operational Research Society*, Vol. 44, No. 1, pp.19–28.

Lee, C.Y. and Kim, S.J. (1995) 'Parallel genetic algorithms for the earliness-tardiness job scheduling problem with general penalty weights', *Computers & Industrial Engineering*, Vol. 28, No. 2, pp.231–243.

Lee, Y.H., Bhaskaram, K. and Pinedo, M. (1997) 'A heuristic to minimize the total weighted tardiness with sequence-dependent setups', *IIE Transactions*, Vol. 29, No. 1, pp.45–52.

Liao, C.J. and Juan, H.C. (2007) 'An ant colony optimization for single-machine tardiness scheduling with sequence-dependent setups', *Computers and Operation Research*, Vol. 34, No. 7, pp.1899–1909.

Mahdavi Mazdeh, M., Nakhjavani, A. and Zareei, A. (2010) 'Minimizing total weighted tardiness with drop dead dates in single machine scheduling problem', *International Journal of Industrial Engineering & Production Research*, Vol. 21, No. 2, pp.89–95.

Marichelvam, M.K. (2012) 'An improved hybrid cuckoo search (IHCS) metaheuristics algorithm for permutation flow shop scheduling problems', *International Journal of Bio-Inspired Computation*, Vol. 4, No. 4, pp.200–205.

Noghrehabadi, A., Ghalambaz, M., Ghalambaz, M. and Vosough, A. (2011) 'A hybrid power series – cuckoo search optimization algorithm to electrostatic deflection of micro fixed-fixed actuators', *International Journal of Multidisciplinary Sciences and Engineering*, Vol. 2, No. 4, pp.22–26.

Rubin, P.A. and Ragatz, G.L. (1995) 'Scheduling in a sequence dependent setup environment with genetic search', *Computers and Operations Research*, Vol. 22, No. 1, pp.85–99.

Tan, K.C. and Narasimhan, R. (1997) 'Minimizing tardiness on a single processor with sequence dependent setup times: a simulated annealing approach', *Omega*, Vol. 25, No. 6, pp.619–634.

Tan, K.C., Narasimhan, R., Rubin, P.A. and Ragatz, G.L. (2000) 'A comparison of four methods for minimizing total tardiness on a single processor with sequence dependent setup times', *Omega*, Vol. 28, No. 3, pp.313–325.

Tasgetiren, M.F., Pan, Q.K. and Liang, Y.C. (2009) 'A discrete differential evolution algorithm for the single machine total weighted tardiness problem with sequence dependent setup times', *Computers and Operations Research*, Vol. 36, No. 6, pp.1900–1915.

Valente, J.M.S. and Alves, R.A.F.S. (2008) 'Beam search algorithms for the single machine total weighted tardiness scheduling problem with sequence dependent setups', *Computers and Operations Research*, Vol. 35, No.7, pp.2388–2405.

Valente, J.M.S. and Schaller, J.E. (2010) 'Improved heuristics for the single machine scheduling problem with linear early and quadratic tardy penalties', *European Journal of Industrial Engineering*, Vol. 4, No. 1, pp.99–129.

Valian, E., Mohanna, S. and Tavakoli, S. (2011) 'Improved cuckoo search algorithm for feed forward neural network training', *International Journal of Artificial Intelligence & Applications*, Vol. 2, No. 3, pp.36–43.

Vallada, E., Ruiz, R. and Minella, G. (2008) 'Minimizing total tardiness in the m-machine flowshop problem: a review and evaluation of heuristics and metaheuristics', *Computers and Operations Research*, Vol. 35, No. 4, pp.1350–1373.

Yang, X.S. and Deb, S. (2009) 'Cuckoo search via Lévy flights', *NaBIC 2009: Proceedings of the World Congress on Nature & Biologically Inspired Computing*, Coimbatore, India, pp.210–214.

Yang, X.S. and Deb, S. (2010) 'Engineering optimisation by cuckoo search', *International Journal of Mathematical Modelling and Numerical Optimisation*, Vol. 1, No. 4, pp.330–343.

Yang, X-S. and Deb, S. (2011) 'Multiobjective cuckoo search for design optimization', *Computers and Operations Research*, Vol. 4, No. 6, pp.1616–1624.

Ying, K., Lin, S. and Huang, C. (2009) 'Sequencing single machine tardiness problems with sequence dependent setup times using an iterated greedy heuristic', *Expert Systems with Applications*, Vol. 36, No. 3, pp.7087–7092.

Zhou, H., Cheung, W. and Leung, L.C. (2009) 'Minimizing weighted tardiness of job-shop scheduling using a hybrid genetic algorithm', *European Journal of Operational Research*, Vol. 194, No. 3, pp.637–649.