A Crossover Developed Ant Colony Optimization for an Integer Partial Flexible Open Shop Scheduling Problems

G. Umashankar[1], M. Nagamani[2], Dr. D. Saravanan[3]

[1] Research Scholar, Bharathiyar University, Coimbatore, India.
[2] Professor of Mathematics, Global Institute of Engineering & Technology, Vellore, Tamil nadu, India.
[3] Professor of Mathematics, Karpaga Vinayagakar College of Engineering, Chengalpattu, Tamil nadu, India.

Abstract: In this paper, we propose a crossover developed ant colony (CDAC) optimization for solving the multi-objective integer partial flexible open shop scheduling problem. In the crossover algorithm, every nourishment sources is givenby two vectors, i.e.,the machine task vector and the operationschedulingvector. The developed ant is divided into three groups, workerants, spectator, and scouts ants. Furthermore, an external developed archive set is introduced to record non-dominated solutions found so far. To balance the exploration and exploitation capability of the algorithm, the scout ants in the crossover algorithm are divided into two parts. The scout ants in single part perform arbitrarily look in the predefined areawhile each scout ant in another part randomly choose one non-dominated solution from the developed archive set. Experimental results on the notable benchmark instances and comparisons with other recently demonstrated algorithms show the proficiency and effectiveness Of the proposed algorithm.

Keywords: *Partial flexible open shop scheduling problem, developed ant colony, multi-objective Optimization, hybrid algorithm.*

1. Introduction

The flexible open shop scheduling problem (FOSP), as a part of the established open shop scheduling problem (OSP), has been considered in very late years. Brandimarte (1993) [1] is among the main author to unravel the FOSP occasions with tabu search (TS) algorithm. In very recent years, some meta-heuristic algorithms, such as TS algorithm [2] [3], particle swarm optimization (PSO) [4] [5], ant colony optimization (ACO) [6], and genetic algorithm (GA) [7] [8], have been utilized in illuminating the single-objective FOSPs. In spite of the fact that the single-objective FOSP has been broadly examined, the research on the multi-objective FOSP is as yet thought to be relative constrained. Kacem et al. (2002a, 2002b)[9] [10] proposed an effective evolutionary algorithm. Xia and Wu(2005) [11] studied the problem with the hybrid algorithm of the PSO and the simulated annealing (SA). Zhang et al. (2009) [12] presented a hybrid algorithm combining PSO algorithm with TS algorithm. Ho et al. (2008) [13]examined a hybrid evolution algorithm joined with a guided local search and an external Pareto chronicle set.

In this paper, we propose a crossover algorithm combining an external Pareto chronicle set and the artificial ant colony (AAC) optimizer to solve the multi-objective FOSP. Whatever is left of this paper is composed as pursues: In Section 2, we briefly depict the problem formulation. Then, the artificial ant colony (AAC) algorithm is introduced in Section 3. The components and framework of the crossover algorithm are exhibited in Section 4 while Section 5 demonstrates the experimental results and comparisons with other algorithms in the literature to demonstrate the superiority of the CAAC performance. At last, the last section presents conclusion of our work.

2. Problem Formulation

The FOSP considers *n* jobs to be processed on *m* machines. There are some assumptions and constrains in the FOSP considered in this study as follows: 1) each job has predefined number of operations and a known determined sequence among these operations; 2) each machine and each operation is ready at zero time; 3) each machine can just proceed one operation at a moment candidate machine set instead of only one machine like in JSP;4) each machine can process another operation simply subsequent to finishing the forerunner activity; 5) every operation can be worked on guaranteed 6) given an operation *Oij*and the selected machine *Mk*, the processing time *pijk*is also fixed.

Given *Ci* chance to be completion date of job *Ji*. *Wk*is the workload of machine *Mk*, which is the aggregate processing time of operations that are worked on machine *Mk*. *Pijk*be the processing time of *Oij*on machine *Mk*. Three destinations are considered in this investigation, specifically [13]:

1) Minimization of maximum completion time (makespan):

$$F_1 = max\{C_i / i = 1, \ldots, n\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

2) Minimization of total workload

$$F_2 = \Sigma p_{i,j,k} \ldots\ldots\ldots\ldots\ldots\ldots (2)$$

3) Minimization of critical machine workload:

$$F_3 = max\{W_k | \ k = 1, \ldots, m\} \ldots\ldots\ldots\ldots\ldots\ldots (3)$$

3. Artificial Ant Colony Algorithm

Very recently, by simulating the behavior of honey bee swarm intelligence, an efficient antcolony (AAC) algorithm is proposed by Karaboga ([14] - [17]). Because of its simplicity and ease of implementation, the AAC algorithm has increased increasingly consideration and has been utilizedto tackle numerous practical engineering problems. In the basic AAC algorithm ([14] - [17]), there are two components: the foraging artificial bees and the food source. The situation of a food source represents a conceivable solution to the advancement problem and the food measure of a food source compares to the quality or fitness of the related solution. The artificial antis segregated into three segregates, worker ants, spectators, and scouts ants. The workerant is one type of ant who is present performing misuse on a food source. A ant that is sitting tight in the hive for settling on choice to pick a food source is called an spectators. The scout ant is a ant who perform investigation procedure and irregular misuse hunt to find a new food source. The main fundamentals of the algorithm are given as follows ([14] - [18]).

Step1. *Deliver initial population;*

Step2*. While stop criteria is not fulfilled, perform steps 3 to steps 6.*

Step3. *Send the worker ants onto their food sources.*

Step4. *Send the spectator ants onto the food sources relying upon their food measure.*

Step5. *Send the scout ants to look conceivable new food sources.*

Step6.*Retainthe best food source found up until now.*

4. The Crossover algorithm CAAC

The fundamental AAC algorithm was initially intended for persistent capacity optimization. In order to make it relevant for solving the problem considered, a novel crossover version of theAAC algorithm, named CAAC, is proposed in this section.

Solution representation

The arrangement of the problem is given with two vectors [19]: the machine taskvector and the operation scheduling vector. The first part places the doled out machine number for each operation at the corresponding position, while the second part puts the same number symbol for each operation of a job and translate them as per the event in the operation scheduling vector.

Worker Ant phase

The worker ant is to play the local search around a given food source. In this way, the worker ant takes the misuse hunt of the algorithm. In order to create good quality and decent varietyneighboring solutions, two sorts of local search operators are connected for the workerants in this study, which are shown as follows.

1. **Local pursuit operator in machine task component**

The local pursuit operator in machine task component is extremely basic and simple to be actualized.The annoyance is obtained by following steps.

Step1. Select a situation in the machine task component, arbitrarily or utilizing some priority rules.

Step2. Assign a suitable machine different with the old one for the operation in the corresponding position.

Step3. Supplant the machine number in the selected situation and create the new machine task component for the solution.

2. **Local pursuit operator in operation scheduling component**

The local pursuit in the operation scheduling component is much the same as the annoyancein solving the JSP, where embedand swap operations are usually utilized in the literature [20-22]. The embed operator is to evacuate a number symbol for an operation in the annoyanceπ from its unique position j and embed it into another position k such that ($k \neq j$). The swap operator is to trade two job symbols of π in the distinctive positions.

After playing out the above two local pursuit approaches, the worker bee acquires a new neighboring food source around the old one. Then the new food source will be evaluated and compared with the old one. The better food source will be kept in the population as in the essential AAC algorithm which plays out a greedy selection procedure.

Spectator Ant phase

In the traditional AAC algorithm, every spectator bee chooses a food source based on the percent of the food measure of every food source among the aggregate food sums. However, the above approach expands extensive computational time to registerthe food measure of each food source.

Hence, we propose a competition selection with the span of 3 in the CAAC algorithm.

In the competition selection, three food sources are picked arbitrarily from the population, and then the food source with most elevated food measure will be selected by the spectator ant. Subsequent toselecting the food source, every spectator ant performs local pursuit for the selected food source and create a new neighboring food source. The better food source between the old one and the new neighboring one will be remembered in the population.

Scout Ant phase

A scout ant performs haphazardly seek in the fundamental AAC algorithm. This will expand the population diversity and keep away local minima, while this will likewisediminish the search efficacy.Since the food sources remembered in the Pareto archive set frequently convey preferable data overothers and the pursuit space around these non-dominated solutions could be the most encouragingregion. Therefore, in the CAAC algorithm, the scout ants are first divided into two parts. One portion of the scout bees haphazardly select a solution from the outer Pareto archive set and play out few embed and swap operators to the selected solution, while the other half scout bees perform randomly look in the predefined seek scope. In the crossover algorithm, not less than5% − 10% ofthe population is scout ants.

Multi-objective optimizer

The Pareto archive set AS

To furnish a set of arrangements with good assorted variety, a Pareto archive set (AS) was presented in this investigation, which is utilized to keep up a limited number of non-dominated arrangements found up until now. Amid the optimization process, the archive set is iteratively refreshed with including some non dominatedarrangements and evacuating some dominated arrangements to get nearer to the Pareto-optimal front. Once another non-dominated arrangement is discovered, it will be added to AS and any arrangementwhich is dominated by the additional one will be expelled from AS. On the off chance that AS becomes overfull, its member which is in the crowded domain is dispensed to maintain the assorted variety of the Pareto archive set.

The storage structure of AS

To reduce the computational time complexity consumed on the update process of the archive set, the members of the AS firstly sequence in an ascending order according to their first objective function value (Pan, 2009) [21].

Non-dominated arranging algorithm

For the population, we should sequence every arrangement as per a specific criteria. For multi-objective optimization problems, we can't utilize one objective function value to decidethe arrangement quality. In this study, a non-dominated arrange algorithm (Deb et al., 2002) [23]was introduced to segregates the population arrangements into few levels as per their dominated arrangements number.

The framework of CACC

The details steps of the proposed CACC algorithm are as per the following:

Step1 Initialization phase;

Step 1.1 Set the system parameters;

Step 1.2 Produce the underlying population.

Step2 Apply the Pareto non-dominated sorting function on the population, and then update the external Pareto archive set by using the solutions in the first Pareto level front.

Step3 If the stopping criterion is satisfied, output the non-dominates solutions in the external Pareto archive set; otherwise, perform steps 4-7.

Step4 Worker ant phase.

Step 4.1 Put each worker ant on every arrangement in the population.

Step 4.2 For each worker ant, perform local search on the appointed arrangement and createanother new neighboring arrangement.

Step 4.3 Evaluate the new neighboring arrangement and record the better arrangement among the new arrangement and the old one as the present arrangement and place it into the population.

If the two arrangements are non-dominated with oneanother, arbitrarily select one as the present arrangement.

Step 4.4 If a arrangement has not been enhanced through cutoff cycles, at that point the comparingworker ant turns into a scout bee and perform step 6.

Step 4.5 Evaluate every arrangement comparing to each worker ant, apply the Pareto non-dominated arranging algorithm on the new population and refresh the outsidePareto archive set utilizing the arrangements in the main Pareto level.

Step5 Spectator ant phase.

Step 5.1 For every spectator ant, haphazardly chooses three arrangements from thepopulation and selects the best one as the food source. In the event that the three arrangements can't command oneanother, then haphazardly select a non-dominate arrangement.

Step 5.2 For each spectator ant, performs local search for the selected food source and carries over the greedy selection procedure to record the better arrangement in the population.

Step 5.3 Evaluate every arrangement comparing to each spectator ant, apply the Pareto non-dominated arranging algorithm on the new population and refresh the outsidePareto archive set using the solutions in the first Pareto level.

Step6 Scout ant phase.

Step 6.1 Divide the scout ants into two sections with a similar number of ants.

Step 6.2 The scout bees in the first part randomly select a food source and perform local search operator in the predefined region. After generating a new solution, performs greedy selection procedure.

Step 6.3 Each scout ant in the second part arbitrarily select a non-dominate arrangement in the outside Pareto archive set and perform few local search for the selected arrangement..After creating a new arrangement, performs greedy selection procedure.

Step 6.4 Evaluate every arrangement comparing to each scout ant, apply the Pareto nondominated arranging algorithm on the new population and update the outside Pareto archive set utilizing the arrangements in the primary Pareto level.

Step7 go to step 3.

5. Experiment Results

This section describes the computational experiments to evaluate the performance of the proposedalgorithm. The test samples come from Kacem instances set [9]. The current instantiation was implemented in C++ on a Pentium IV 1.8GHz with 512M memory.

5.1 Setting parameters

Each example can be portrayed by the accompanying parameters: number of jobs (n), number of machines (m), and the number of operations (*optimum*). Followings are the detail parameters esteem:

The extent of the population is equivalent to the quantity of workerant and the quantity of spectator, which is set to $5n$; the maximum cycle of the algorithm is set to $10\times n\times m$; the point of confinement number of cycles through which no change happens on the food source, at that point worker ant becomes a scout ant; the utmost number is set to $n\times m2$; the percent of scout ant is set to an arbitrary number at the range of 0.05 and 0.1.

5.2 Results comparisons

The five test instances come from Kacem [9] [10], which range from 4 jobs×5 machines to 15 jobs ×10 machines. Two tests are performed for comparison, i.e. the instances with singleobjective and the problems with three objectives. Several recently published algorithms are compared with the proposed CAAC algorithm, such as the AL+CGA proposed by Kacemetal. (2002b) [10] ,the GENACE approach utilized by Ho (2004) [24], the PSO+SA createdby Xia and Wu (2005) [11], the ant systems & local search optimization technique (hereafter called ACO+LS) exhibited by Liouane et al. (2006) [6], and the PSO+TS presented by Zhang (2009) [12]. The single objective five Kacem instancesFor solving the five instances with single objective to minimize the makespan criterion, the experimental results and comparisons are given in Table 1. It can be seen from Table 1 that the CAAC algorithm can acquire the best results for all the Kacem instances. The proposed algorithm outperforms the AL+CGA in 4 out of 5 instances, while outperforms the GENACE technique in 2 out of 4 cases. For correlation with the very recently distributed algorithms, the CAAC algorithm acquired a superior result in tackling the largest problem than the ACO+LS proposed by Liouane (2006). Particle swarm optimization (PSO) is a proficient swarm intelligent algorithm and the experimental results acquired by PSO+SA and PSO+TS are considered as the aggressive results for the FJSP. Table 1 gives that the CAAC algorithm outperforms the PSO+SA algorithm in 2 out of 3 problems. The CAAC algorithm can get the same experimental results with the PSO+TS in short computational times. For instance, in tackling the largest problem 15 × 10, our algorithm devours pretty much 50 seconds to achieve the best result up until now.

Problem set	AL+CGA	GENACE	ACO+LS	PSO+SA	PSO+TS	CACO
4×5	16	11	11	-	-	10
8×8	15	-	-	15	14	12
10×7	15	12	11	-	-	11
10×10	7	7	7	7	7	7
15×10	23	12	12	12	11	10

Table 1: Comparison of the five instances with single objective (makespan)

Problem set		AL+CGA		PSO+SA		PSO+TS		CACO		
4×5	F1	16				11		11	11	12
	F2	34				32		31	31	32
	F3	10				10		9	7	6
8×8	F1	15	16	15	16	14	15	13	14	15
	F2	79	75	75	73	77	75	74	73	72
	F3	13	13	12	13	12	12	11	11	12
10×7	F1							11	12	
	F2							60	59	
	F3							10	11	
10×10	F1	7		7		7		7	6	7
	F2	45		44		43		40	41	42
	F3	5		6		6		6	5	4
15×10	F1	23	24	12		11		11	10	
	F2	95	91	91		91		90	92	
	F3	11	11	11		11		10	11	

Table 2 The comparison of the results on the five multi-objective FOSP instances

The multi objective five instances Table 2 shows the comparison of the results on the five multi-objective FOSP instances. Thethree objectives are considered simultaneously, i.e. minimization of the makespan (denoted by*f*1),the total workload (denoted by *f*2), and the maximal workload (denoted by *f*3). It can be seen from Table 2 that the CAAC algorithm is aggressive to different algorithms. The experimentalresults of the proposed algorithm command the consequences of the AL+CGA for unraveling the four examples. For comparison with the very recently distributed algorithms, the CAAC can either acquire more non-dominated solutions or acquire superior result than PSO+TS and PSO+SAalgorithms. For instance, our algorithm acquire three non-dominated

solutions in illuminating the8 × 8 instance,while the PSO+SA and PSO+TS can just get two results. Likewise, our algorithm gets all these results in a run while the other algorithms can acquire only single result in a run. In other words, the other algorithm should run several times to get different results for an instance. In this manner, the outside Pareto archive can develop the population diversity ofour algorithm. To make a further correlation with the ACO+LS proposed by Liouane (2006), we likewise test the problem recorded in the paper [6]. The problem is given in Table 3. The correlation of the experimental results from ACO+LS and our algorithm are given in Table 4 and the two arrangementsacquired by the CAAC algorithm are given in Table 5 and Table 6, separately. It very well may be seen from Table 4 that the CAAC algorithm got two non-dominated solutions for the example benchmark while the TS method in literature [6] can getonly one solution. Besides, the resulted solutions acquired by our algorithm ruled all the results by the ACO technique.Furthermore, our algorithm acquired the two non-dominated solutions expending simply 0.01 seconds for the instance benchmark. Therefore, Table 4 concludes that the proposed algorithm is efficient in solving the example problem especially when compared with the ACO method.

		M1	M2	M3	M4	M5	M6
J1	O11	10	7	6	13	5	1
	O12	4	5	8	12	7	11
	O13	9	5	6	12	6	17
	O14	7	8	4	10	15	3
J2	O21	15	12	8	6	10	19
	O22	9	5	7	13	14	7
	O23	14	13	14	20	8	17
J3	O31	7	16	5	11	17	9
	O32	9	16	8	11	6	3
	O33	6	14	8	18	21	14

Table 3: Example benchmark 3 jobs-6 machines

	F1	F2	F3	Avgtime(s)
Lower bound value	18	45	8	
ACO+LS	19	51	13	
	19	50	13	
	19	48	14	
	19	47	14	
TS	18	45	12	-
CACO	17	44	11	0.01
	18	45	9	

Table 4: Comparison of the example benchmark

	O1	O2	O3	O4
J1	M6 [0,1]	M1 [1,5]	M2 [5,10]	M6 [10,13]
J2	M4 [0,6]	M5 [6,10]	M5 [10,18]	***
J3	M3 [0,5]	M6 [5,8]	M1 [8,14]	***

Table 5: Solution 1 for the example benchmark (f1=18, f2=45, f3=12)

	O1	O2	O3	O4
J1	M6 [0,1]	M1 [1,5]	M2 [11,16]	M6 [16,19]
J2	M4 [0,6]	M5 [6,11]	M5 [11,19]	***
J3	M3 [0,5]	M6 [5,8]	M1 [8,14]	***

Table 6: Solution 2 for the example benchmark (f1=19, f2=46, f3=10)

6. Conclusions

In this paper, we have proposed an effective algorithm for solving multi-objective FOSPs. Rather than applying the fundamental AAC algorithm, we build up a crossover AAC method. To remember the non-dominated solutions found up until now and increment the population diversity, we exhibited an outside Pareto archive set. A quick Pareto update function is additionally introduced in the algorithm to improve the computational capacity. In the crossover algorithm, the equalization of the capacity of investigation and misuse is considered.

Experimental results on several well-known benchmarks show that our algorithm is aggressive to other recently distributed algorithms for solving the FOSPs. The future work is to develop the neighborhood structure of the problem considered and improve the assembly capacity of the algorithm.

Reference:

[1] P. Brandimarte. Routing and scheduling in a flexible job shop by tabusearch.Annals ofOperations Research , Vol 22, pp. 158-183, 1993.

[2] M. Mastrolilli and L. M. Gambardella. Effective neighborhood functions for the flexible jobshop problem.Journal of Scheduling , Vol. 3(1), pp. 3-20, 2000.

[3] J. Q. Li, Q. K. Pan, P. N. Suganthan and T. J. Chua. A hybrid tabu search algorithm with anefficient neighborhood structure for the flexible job shop scheduling problem. InternationalJournal of Advanced Manufacturing Technology. doi: 10.1007/s00170-010-2743-y.

[4] L. Gao, C. Y. Peng, C. Zhou and P. G. Li. Solving flexible job shop scheduling problemusing general particle swarm optimization. In Proceedings of the 36th CIE Conference onComputers & Industrial Engineering, pp. 3018-3027, 2006.

[5] J. Q. Li, Q. K. Pan and S. X. Xie. A hybrid variable neighborhood search algorithm forsolving multi-objective flexible job shop problems. ComSIS Computer Science of SoftwareInformation and System. doi: 10.2298/CSIS090608017L.

[6] N. Liouane, I. Saad, S. Hammadi and P. Borne. Ant Systems & Local Search Optimizationfor Flexible Job-Shop Scheduling Production. International Journal of Computers, Communications & Control, Vol. 2, pp. 174-184, 2007.

[7] J. Gao, L. Sun and M. Gen. A hybrid genetic and variable neighborhood descent algorithmfor flexible job shop scheduling problems.Computers& Operations Research, Vol. 35(9), pp.2892-2907, 2008.

[8] F. Pezzella, G. Morganti and G. Ciaschetti. A genetic algorithm for the Flexible Job-shop Scheduling Problem.Computers& Operations Research , Vol. 35, pp. 3202-3212, 2008.

[9] I. Kacem, S. Hammadi and P. Borne. Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic. Mathematics andComputers in Simulation, Vol. 60, pp. 245-276, 2002a.

[10] I. Kacem, S. Hammadi and P. Borne. Approach by localization and multi-objective evolutionary optimization for flexible job-shop scheduling problems,IEEE Transactions on Systems, Man and Cybernetics, Part C, Vol. 32(1), pp. 408-419, 2002b.

[11] W. J. Xia and Z. M. Wu. An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems.Computers& Industrial Engineering , Vol. 48(2), pp.

[12] G. H. Zhang, X. Y. Shao, P. G. Li and L. Gao. An effective hybrid swarm optimizationalgorithm for multi-objective flexible job-shop scheduling problem. Computers & IndustrialEngineering, Vol. 56(4), pp. 1309-1318, 2009.

[13] N. B. Ho and J. C. Tay. Solving Multiple-Objective Flexible Job Shop Problems by Evolutionand Local Search. IEEE Transactions on Systems, Man, and Cybernetics, Part C:Applicationsand Reviews , Vol. 38(5), pp. 674-685, 2008.

[14] D. Karaboga. An idea based on honey bee swarm for numerical optimization. TechnicalReport TR06. Computer Engineering Department. Erciyes University. Turkey. 2005.

[15] D. Karaboga and B. Basturk. A powerful and efficient algorithm for numerical functionoptimization: Artificial Bee Colony (ABC) algorithm.Journal of Global Optimization , Vol.39(3), pp. 459-171, 2007.

[16] D. Karaboga and B. Basturk. On The performance of Artificial Bee Colony (ABC) algorithm.Applied Soft Computing , Vol. 8(1), pp. 687-697, 2008.

[17] D. Karaboga and B. Akay. A comparative study of Artificial Bee Colony Algorithm. AppliedMathematics and Computation,Vol. 214, pp. 108-132, 2009.

[18] Q. K. Pan, M. F. Tasgetiren, P. N. Suganthan and T. J. Chua. A discrete artificial beecolony algorithm for the lot-streaming flow shop scheduling problem. Information Sciences.doi: 10.1016/j.ins.2009.12.025, 2010.

[19] J. Q. Li, Q. K. Pan and Y.C. Liang. An effective hybrid tabu search algorithm for multiobjective flexible job shop scheduling problems. Computers & Industrial Engineering, Vol.59, pp. 647-662, 2010b.

[20] F. Pezzella, G. Morganti and G. Ciaschetti. A genetic algorithm for the Flexible Job-shopScheduling Problem. Computers & Operations Research, Vol. 35, pp. 3202-3212, 2008.

[21] Q. K. Pan, L. Wang, B. Qian. A novel differential evolution algorithm for bi-criteria no-waitflow shop scheduling problems.Computers& Operations Research, Vol. 36(8), pp. 2498 -2511,2009.

[22] L. Wang. Shop scheduling with genetic algorithms. Tsinghua university press, Beijing,China, 2003.

[23] K. Deb, A. Paratap, S. Agarwal and T. Meyarivan. A fast and elitist multiobjective geneticalgorithm: NSGA-II. IEEE Transactions on Evolutinary Computation, Vol. 6(2), pp. 182-197,2002.

[24] N. B. Ho and J. C. Tay. GENACE: An Efficient Cultural Algorithm for solving the Flexible Job-Shop Problem. In Proceedings of the IEEE Congress on Evolutionary Computation