A Multi-Objective Ant-Colony Algorithm for Permutation Flowshop Scheduling to Minimize the Makespan and Total Flowtime of Jobs

Chandrasekharan Rajendran[1] and Hans Ziegler[2]

[1] Department of Management Studies
Indian Institute of Technology Madras, Chennai - 600 036, India
craj@iitm.ac.in
[2] Faculty of Business Administration and Economics
Department of Production and Logistics
University of Passau, D-94032 Passau, Germany
ziegler@uni-passau.de

Summary. The problem of scheduling in permutation flowshops is considered with the objectives of minimizing the makespan and total flowtime of jobs. A multi-objective ant-colony algorithm (MOACA) is proposed. The salient features of the proposed multi-objective ant-colony algorithm include the consideration of two ants (corresponding to the number of objectives considered) that make use of the same pheromone values in a given iteration; use of a compromise objective function that incorporates a heuristic solution's makespan and total flowtime of jobs as well as an upper bound on the makespan and an upper bound on total flowtime of jobs, coupled with weights that vary uniformly in the range [0, 1]; increase in pheromone intensity of trails by reckoning with the best solution with respect to the compromise objective function; and updating of pheromone trail intensities being done only when the ant-sequence's compromise objective function value is within a dynamically updated threshold level with respect to the best-known compromise objective function value obtained in the search process. In addition, every generated ant sequence is subjected to a concatenation of improvement schemes that act as local search schemes so that the resultant compromise objective function is improved upon. A sequence generated in the course of the ant-search process is considered for updating the set of heuristically non-dominated solutions. We consider the benchmark flowshop scheduling problems proposed by Taillard (1993), and solve them by using twenty variants of the MOACA. These variants of the MOACA are obtained by varying the values of parameters in the MOACA and also by changing the concatenation of improvement schemes. In order to benchmark the proposed MOACA, we rely on two recent research reports: one by Minella *et al.* (2008) that reported an extensive computational evaluation of more than twenty existing multi-objective algorithms available up to 2007; and a study by Framinan and Leisten (2007) involving a multi-objective iterated greedy search algorithm, called MOIGS, for flowshop scheduling. The work by Minella concluded that the multi-objective simulated annealing algorithm by Varadharajan and Rajendran (2005), called MOSA, is the best performing multi-objective algorithm for permutation flowshop scheduling. Framinan and Leisten found that their MOIGS performed better than the MOSA in terms of generating more heuristically non-dominated solutions. They also obtained a set of heuristically non-dominated solutions for every benchmark problem instance provided by Taillard (1993) by consolidating the solutions obtained by them and the solutions reported by Varadharajan and Rajendran. This set of heuristically non-dominated solutions (for every problem instance, up to

U.K. Chakraborty (Ed.): Comput. Intel. in Flow Shop and Job Shop Sched., SCI 230, pp. 53–99.
springerlink.com

100 jobs, of Taillard's benchmark flowshop scheduling problems) forms the reference or benchmark for the present study. By considering this set of heuristically non-dominated solutions with the solutions given by the twenty variants of the MOACA, we form the net heuristically non-dominated solutions. It is found that most of the non-dominated solutions on the net non-dominated front are yielded by the variants of the MOACA, and that in most problem instances (especially in problem instances exceeding 20 jobs), the variants of the MOACA contribute more solutions to the net non-dominated front than the corresponding solutions evolved as benchmark solutions by Framinan and Leisten, thereby proving the effectiveness of the MOACA. We also provide the complete set of heuristically non-dominated solutions for the ninety problem instances of Taillard (by consolidating the solutions obtained by us and the solutions obtained by Framinan and Leisten) so that researchers can use them as benchmarks for such research attempts.

1 Introduction

Flowshop scheduling problem involves the determination of an order of processing n jobs over m machines, arranged in series, to meet a desired objective or a measure of performance. The static permutation flowshop scheduling problem has been widely investigated over the years by considering separately the objectives of minimizing the makespan and total flowtime of jobs, and with the consideration of developing exact or heuristic methods (e.g. Johnson (1954), Ignall and Schrage (1965), Campbell *et al.* (1970), Gelders and Sambandam (1978), Miyazaki *et al.* (1978), Miyazaki and Nishiyama (1980), Nawaz *et al.* (1983), Rajendran (1993), Ho (1995), Wang *et al.* (1997), Woo and Yim (1998), Liu and Reeves (2001), Chung et al. (2002), Allahverdi and Aldowaisan (2002), Framinan and Leisten (2003), Framinan *et al.* (2005), Ruiz and Stuetzle (2007), Kalczynski and Kamburowski (2007) and (2008), Dong *et al.* (2008), Laha and Chakraborty (2008)). The use of metaheuristics such as simulated annealing, genetic algorithm and tabu search has been frequently resorted to solve flowshop scheduling problems (e.g. Widmer and Hertz (1989), Ben-Daya and Al-Fawzan (1998), and Ruiz *et al.* (2006)). In recent times, attempts are being made to solve combinatorial optimization problems by making use of swarm-intelligence algorithms. An important algorithm in this class is the ant-colony-optimization algorithm (or simply, ant-colony or ACO algorithm). The pioneering work has been done by Dorigo (1992), and an introduction to the ACO algorithms had been dealt with in Dorigo *et al.* (1996). Attempts have been made to solve the permutation flowshop scheduling problem with the objective of minimizing the makespan / total flowtime of jobs using ACO algorithms (e.g. Stuetzle (1998) dealing with the permutation flowshop scheduling problem with the objective of minimizing the makespan; Merkle and Middendorf (2000) dealing with the single-machine scheduling problem; T′kindt *et al.* (2002) considering the two-machine flowshop scheduling problem; and Rajendran and Ziegler (2004) and (2005) considering the m-machine permutation flowshop scheduling problem). Another swarm intelligence algorithm is the particle swarm algorithm which has shown promising results to solve flowshop scheduling problems (e.g., Tasgetiren *et al.* (2007) and Liao *et al.* (2007)).

While many attempts have been made to minimize separately makespan and total flowtime, only some attempts have been made to simultaneously minimize such

measures of performance. In such a case, it is common to develop algorithms to obtain a set of Pareto-optimal solutions (or at least a set of heuristically non-dominated solutions). Two approaches to multi-objective scheduling are widely followed, namely, *a priori* approach in which the objectives are combined in the form of a weighted compromise function (mostly linear), and *a posteriori* approach in which a set of efficient or Pareto-optimal solutions (in the case of optimality being guaranteed) or a set of heuristically-efficient or heuristically non-dominated solutions (in the case of optimality being not guaranteed) is obtained. In the following, the term 'non-dominated solutions' or 'non-dominated sequences' refers to heuristically-efficient or heuristically non-dominated solutions or sequences, without the guarantee of efficiency or Pareto optimality. Some attempts in these directions are due to Rajendran (1994) and (1995), Sridhar and Rajendran (1996), Murata *et al.* (1996), Ishibuchi and Murata (1998), Bagchi (1999), Chang *et al.* (2002), Framinan *et al.* (2002), and Arroyo and Armentano (2005). In addition, attempts have also been done with the consideration of a lexicographical approach of optimizing a set of objectives (e.g., Daniels and Chambers (1990), Rajendran (1992), Chakravarthy and Rajendran (1999), T'kindt *et al.* (2002), Allahverdi (2004), and Framinan and Leisten (2006)).

Varadharajan and Rajendran (2005) developed a multi-objective simulated-annealing algorithm (with two variants, called MOSA-I and MOSA-II) for flowshop scheduling to minimize the makespan and total flowtime of jobs. The MOSA aims at discovering non-dominated solutions through the use of a simple probability function that is varied in such a way that the entire objective space is covered uniformly, thereby obtaining many non-dominated and well-dispersed solutions. The authors considered the benchmark flowshop problems of Taillard (1993), and obtained the non-dominated solution set for every problem, yielded by existing multi-objective flowshop scheduling algorithms, namely, the algorithms by Ishibuchi and Murata (1998), Bagchi (1999), Chang *et al.* (2002), and Framinan *et al.* (2002), as well as those by MOSA-I and MOSA-II. Subsequently they obtained the net non-dominated front by consolidating all the non-dominated fronts. They found that, in most cases, the MOSA contributes the most to the net non-dominated solution set, in comparison to the existing algorithms. Framinan and Leisten (2007) proposed a multi-objective iterated greedy search, called MOIGS, that is based on a partial enumeration heuristic. The MOIGS uses a parameter, called d, and the authors tried with the values of d ranging from 3 to 10. They found that the MOIGS discovers more non-dominated solutions than those discovered by Varadharajan and Rajendran (2005). In addition, they consolidated the solutions yielded by their MOIGS (with different values for d) and the solutions obtained by Varadharajan and Rajendran. It is be noted that Varadharajan and Rajendran consolidated the solutions yielded by MOSA-I, MOSA-II, and the solutions yielded by the algorithms of Ishibuchi and Murata (1998), Bagchi (1999), Chang *et al.* (2002), and Framinan *et al.* (2002). It is therefore evident that the final non-dominated solutions obtained by Framinan and Leisten are drawn from the implementations of MOIGS with eight different values of d, and from the implementations of MOSA-I, MOSA-II, and other algorithms considered by Varadharajan and Rajendran. These non-dominated solutions consolidated by Framinan and Leisten for a problem instance of Taillard (1993) could serve as the benchmark for researchers in the area of flowshop scheduling.

It is to be noted that most of the multi-objective flowshop scheduling algorithms were evaluated by the respective authors by comparing with the previously available literature, and that too, with the related objectives. It also appears that many researchers did not attempt to consider the possible adaptation of the generic multi-objective algorithms (such as NSGA by Srinivas and Deb (1994), SPEA by Zitzler and Thiele (1999), PESA by Corne *et al.* (2000), PESA-II by Corne *et al.* (2001), and NSGA-II by Deb *et al.* (2002)) to flowshop scheduling problems. A recent study by Minella *et al.* (2008) is possibly the first significant attempt to perform a comprehensive analysis by considering a number of flowshop scheduling algorithms (such as the multi-objective genetic algorithm by Murata *et al.* (1996), multi-objective tabu search (MOTS) by Armentano and Arroyo (2004), multi-objective genetic local search by Arroyo and Armentano (2005), MOSA by Varadharajan and Rajendran (2005), multi-objective genetic algorithm by Pasupathy *et al.* (2006), and PILS by Geiger (2007)), and also a number of generic multi-objective algorithms such as the NSGA, SPEA, PESA, PESA-II and NSGA-II. In all, a total of twenty three multi-objective algorithms were considered, and performance analyses were carried out. The authors consolidated the solutions for Taillard's benchmark problem instances. It was found that the MOSA by Varadharajan and Rajendran is the best performer among these twenty three algorithms with respect to multi-objective flowshop scheduling.

In the following, the problem of scheduling in permutation flowshops is considered with the objectives of minimizing the makespan and total flowtime of jobs. We present a multi-objective ant-colony algorithm (MOACA) for obtaining heuristically efficient or heuristically non-dominated solutions. Variants of the MOACA are proposed by varying the values of parameters in the MOACA and also by varying the concatenation of local search or improvement schemes that consider a compromise objective function. We make use of data set containing the benchmark flowshop problems of Taillard (1993) (up to 100 jobs), and generate the non-dominated solutions for every flowshop problem instance by using the different variants of the MOACA and the benchmark solutions consolidated by Framinan and Leisten (2007).

2 Formulation of the Multi-Objective Static Permutation Flowshop Scheduling Problem under Study

The static permutation flowshop scheduling problem consists in scheduling n jobs with given processing times on m machines, where the sequence of processing a job on all machines is identical and unidirectional for each job. In studying flowshop scheduling problems, it is a common assumption that the sequence in which each machine processes all jobs is identical on all machines (permutation flowshop). A schedule of this type is called a permutation schedule and is defined by a complete sequence of all jobs. We also consider only permutation sequences in the following.

Let

t_{ij} processing time of job i on machine j.

D_i due-date for job i.

n total number of jobs to be scheduled.

m total number of machines in the flowshop.

σ ordered set of jobs already scheduled, out of n jobs; partial sequence.

$q(\sigma, j)$ completion time of partial sequence σ on machine j (*i.e.* the release time of machine j after processing all jobs in partial sequence σ).

$q(\sigma i, j)$ completion time of job i on machine j, when the job is appended to partial sequence σ.

For calculating the start and completion times of jobs on machines in permutation flowshops, recursive equations are used as follows.

Initialize $q(\sigma i, 0)$, the completion time of job i on machine 0, equal to zero. This time indicates the time of availability of a job in the flowshop, and it is equal to 0 for all jobs in case of static flowshops.

For $j = 1$ to m do

$$q(\sigma i, j) := \max \{ q(\sigma, j) ; q(\sigma i, j\text{-}1)\} + t_{ij}. \tag{2.1}$$

The flowtime of job i, C_i, is given by

$$C_i = q(\sigma i, m). \tag{2.2}$$

It is to be noted that $q(\phi, j)$ is equal to 0 for all j, where ϕ denotes a null schedule.

When all jobs are scheduled, the total flowtime F and the makespan C_{max} of jobs are obtained as follows:

$$F = \sum_{i=1}^{n} C_i \ , \tag{2.3}$$

and

$$C_{max} = \max \{ C_i , i = 1, 2, \ldots , n \}. \tag{2.4}$$

The objective is to simultaneously minimize F and C_{max}. Exceptions set aside, there exists no single solution minimizing both objectives simultaneously. An optimal solution then must have the property of non-dominance.

To present in brief the principle of non-dominance in the context of the problem under study, let us assume that the makespan and total flowtime of jobs yielded by sequence S are denoted by $C_{max}(S)$ and $F(S)$ respectively. For the sake of generality, we let $Z_1(S)$ and $Z_2(S)$ denote $C_{max}(S)$ and $F(S)$ respectively. Sequence S is said to dominate S' if $Z_r(S) \leq Z_r(S') \ \forall \ r$, and $Z_r(S) < Z_r(S')$ for at least one r. Sequence S'' is efficient if there exists no other sequence S dominating S''. It is to be noted that the m-machine permutation flowshop scheduling problem with the consideration of a single objective, in most cases, was shown by Garey *et al.* (1976) to be NP-hard. It is therefore evident that researchers develop heuristic methods to obtain heuristically non-dominated solutions (without the guarantee of efficiency) in the case of multi-objective flowshop scheduling problems. A Sequence S'' is called heuristically non-dominated or heuristically efficient with respect to a given set of heuristic solutions if there exists no other known heuristic sequence S dominating S''. Suppose we have a

set of heuristically non-dominated sequences, denoted by ψ. A new heuristic sequence S'' qualifies for entry into ψ if and only if for each sequence S in ψ there exists at least one r for which $Z_r(S'') < Z_r(S)$. Likewise, a sequence S' can be eliminated from the set ψ due to the inclusion of S'' if $Z_r(S'') \leq Z_r(S') \ \forall \ r$. Readers may see T'kindt and Billaut (2002) for a complete treatment on scheduling with multiple objectives. In the following, the a posteriori approach is considered, i.e., a set of heuristically efficient sequences with respect to the two objectives of minimizing total flowtime and minimizing makespan is to be determined.

3 Description of the Proposed Multi-Objective Ant-Colony Algorithm

3.1 General Structure of Ant-Colony Algorithms

ACO algorithms make use of simple agents, called ants, that iteratively construct solutions to combinatorial optimization problems. The solution generation or construction by ants is guided by (artificial) pheromone trails and problem-specific heuristic information. In the context of combinatorial optimization problems, pheromones indicate the intensity of ant-trails with respect to solution components, and such intensities are determined on the basis of the influence or contribution of each solution component with respect to the objective function. An individual ant constructs a complete solution by starting with a null solution and iteratively adding solution components until a complete solution is constructed. Typically, solution components which are part of better solutions used by ants over many iterations receive a higher amount of pheromone, and hence, such solution components are more likely to be used by the ants in future iterations of the ACO algorithm. This is enhanced by also making use of pheromone evaporation in updating trail intensities. In the context of application of ACO algorithms to scheduling problems, pheromone trail intensity (or desirability) of placing job i in position k of a sequence can be denoted by τ_{ik}. It is to be noted that for every job i for any possible position k, a pheromone value is stored and updated in each iteration of the ACO algorithm. An explanation on the structure of ACO algorithms is given in Stuetzle (1998), and Rajendran and Ziegler (2004).

3.2 Details of the Proposed Multi-Objective Ant-Colony Algorithm (MOACA)

We highlight the salient features of the proposed algorithm with respect to the search in the two-dimensional objective-function space enabled through the use of a compromise objective function incorporating relative weights for each objective function and the use of upper bounds on the makespan and total flowtime of jobs.

3.2.1 Characterization of the MOACA

In view of two objectives being considered, two seed sequences are used corresponding to every combination of the two relative weights related to the makespan and total flowtime of jobs, and these sequences are used to initialize the pheromone trail intensities τ_{ik}. A front that consists of non-dominated sequences

obtained during the search process is maintained. The trail intensities and the best sequence obtained so far are used as the basis to construct multiple (in our study, two) ant sequences which are subsequently improved, with respect to the compromise objective function, by using different concatenations of two local search schemes, called JIS and JSS. We construct two ant sequences in view of the number of objectives being two; moreover, pilot runs with the construction of a greater number of multiple ant sequences have indicated the best performance of the proposed algorithm with two ant sequences, given our restriction on the total number of sequences enumerated in the MOACA. It is to be noted that every ant sequence that is generated (including every sequence generated in local search schemes) is checked for possible entry into the non-dominated front, so as to discover as many solutions lying on the multi-modal non-dominated front as possible.

In the MOACA, we define a compromise objective function for a given sequence S as follows:

$$Z(S) = w_1 \times (C_{max}(S) / up\_C_{max}) + w_2 \times (F(S) / up\_F), \tag{3.1}$$

where up_C_{max} refers to an upper bound on the makespan for a given problem, up_F refers to an upper bound on total flowtime of jobs, and $w_1+w_2 = 1$ with $w_1, w_2 \geq 0$. This approach of using a compromise objective function with the incorporation of upper bounds on the makespan and total flowtime of jobs has been found to be effective in the case of multi-objective flowshop scheduling; the reason is that we basically normalize a heuristic solution's makespan and total flowtime of jobs, thereby avoiding the inconsistency in the magnitude of the makespan and total flowtime of jobs. In fact, similar approaches were also taken by Rajendran (1994) and (1995), and also by Sridhar and Rajendran (1996).

Note that to start with, for a given Taillard's problem instance, we use the upper bound on makespan that was reported by Taillard (1993) (denoted by *upmake* for a given problem instance), and we use the best upper bound on total flowtime that was reported by Rajendran and Ziegler (2004) (denoted by *upflow* for a given problem instance). Initialize $up\_C_{max} = upmake$, and $up\_F = upflow$. However, during the execution of the MOACA, better upper bounds, if obtained, are used to update up_C_{max} and up_F for their use in the compromise objective function. Note that the weights have to be appropriately chosen in order to discover many non-dominated solutions. We vary w_1 uniformly (and consequently w_2) in the range [0, 1]. In the MOACA, we initially set $w_1 = 0$, implying that we first seek to minimize total flowtime of jobs, and we increase w_1 in steps of 0.1, up to 1. This means that the basic MOACA is repeated 11 times, corresponding to different values of w_1 and w_2, and our experimental investigations have shown that the MOACA with these values for weights is able to discover many solutions lying on the non-dominated front (with no possible guarantee of Pareto optimality or efficiency). We now present the mechanics of the basic MOACA, for the given w_1 and w_2, in Sections 3.2.2, 3.2.3 and 3.2.4.

3.2.2 Generation of Two Initial Ant Sequences and Initialization of Trail Intensities

We generate one seed sequence by ordering jobs in the ascending order of the weighted sum of process times of jobs (i.e., in the non-decreasing order of

$\sum_{j=1}^{m}(m-j+1)t_{ij}$; see Rajendran (1993) for details), followed by the improvement scheme presented by Nawaz / Enscore / Ham (1983) if w_1 is less than or equal to 0.5, or by ordering jobs in the non-increasing order of the sum of process times of jobs, and then using the improvement scheme presented by Nawaz / Enscore / Ham (1983) if w_1 is greater than 0.5. Note that all partial and complete sequences (generated during these procedures) are evaluated by using Eq. (3.1), and the best partial (or complete) sequence is accordingly chosen. The second seed sequence is generated randomly by selecting the job to be placed in position k of the sequence with equal probability from the set of unscheduled jobs, $k = 1(1)n$. Check if each complete sequence can enter the existing non-dominated front. If so, enter it and update the front accordingly. Every seed sequence is subjected to the improvement schemes, namely, the job-index-based insertion scheme (called JIS), followed by the job-index-based swap scheme (JSS) in the given concatenation, with the consideration of $Z(S)$ for the given w_1 and w_2 (see Eq. (3.1)). The details of different concatenations of the JIS and JSS, namely, JIS-JSS-JIS-JSS, JIS-JIS-JSS-JIS and JIS-JIS-JIS-JSS, are presented later. The effectiveness of concatenation of the local search schemes is due to the fact that each of these schemes perturbs the seed sequence in different ways, thereby discovering many more local minima in the neighborhood than a single local search scheme applied more than once. These improvement schemes have been found to be effective in single-objective flowshop scheduling by earlier works as well (see Rajendran and Ziegler (2004) and (2005)). In fact, our computational experiments have also shown that the concatenation of such local search schemes has been found to perform better than the successive application of one single local scheme in terms of discovering many more solutions on the non-dominated front. The details of the JIS and JSS are given in the Appendix. Note that the JIS involves a relatively mild perturbation of the seed sequence, as opposed to the JSS. In fact, the JIS can be considered as an intensification of local search, while the JSS can be considered as a diversification of local search. It is also to be noted that each of the two local search schemes aims at improving the seed sequence with the consideration of the compromise objective function (as given in Eq. (3.1)) for the given w_1 and w_2, and that every sequence that is generated in a local search scheme is considered for possible entry in the non-dominated front. The two final sequences that are yielded by the application of concatenation of JIS and JSS on each of the two seed sequences are taken as the final ant sequences. These sequences, denoted by S^1 and S^2, are used to set the trail intensities for a given w_1 and w_2. Let these two sequences' compromise objective function values (computed by using Eq. (3.1)) be denoted by $Z(S^1)$ and $Z(S^2)$ respectively. Let the minimum of these two values be denoted by Z^*, and the corresponding sequence be denoted by S^*. We initialize pheromone trail intensities as follows:

$$\tau_{ik} = 1/(Z^*)^p, \forall\ k \text{ and } \forall i. \tag{3.2}$$

In the above, p (>= 1) denotes the index of power. Initialize *no_iter*, the number of iterations in respect of generation of ant sequences in the search process for the given w_1 and w_2, to 0. Subject S^1 and S^2 to an adjacent pairwise interchange of jobs (interchanging jobs found in positions k and k+1, for k = 1, 2, ..., n-1), thereby generating 2(n-1) sequences in the neighbourhood. Check every generated sequence for possible entry into the non-dominated front and also check for the consequent

updating of the non-dominated front. Note that these 2(n-1) sequences do not have any impact on trail intensities, and that these sequences are generated to primarily explore the neighbourhood for non-dominated solutions.

3.2.3 Modification of Trail Intensities

We first modify the trail intensities as follows:

$$\tau_{ik} := \rho \times \tau_{ik} \text{ , } \forall \ k \text{ and } \forall i, \tag{3.3}$$

where ρ denotes the persistence rate of pheromone trail intensities (or equivalently, 1-evaporation rate).

Then, we further modify the trail intensities τ_{ik} as follows, by taking into account the position occupied by a job.

For r = 1 and 2, do the following: /*corresponding to two sequences*/
  if (($Z(S^r) - Z^*$) / Z^*) ≤ *cut_off*
  then
    for i = 1(1) n do the following: /*corresponding to n jobs*/
      for k = 1(1) n do the following: /*corresponding*/
        /*to n positions*/
        if | $h^r(i)$ - k | ≤ $\lfloor n/50 \rfloor$
        then
          set $\tau_{ik} := \tau_{ik} + 1 / (2 \times (Z(S^r))^p)$. (3.4)

In the above, *cut_off* refers to the threshold value with respect to the deviation of the compromise objective function value of a given sequence from the best value obtained so far in the MOACA, for the given w_1 and w_2. If the deviation is less than or equal to the *cut_off*, then we use the sequence to update the trail intensities. This is done so because we do not want to use an inferior sequence to be used in updating pheromone values, as otherwise, we would lose the good trail intensities that have been obtained in the search process. In the above, $h^r(i)$ refers to the position occupied by job i in sequence S^r. It is to be noted that we update the trail intensity for every job with respect to more than one position, depending upon the value of $\lfloor n/50 \rfloor$. This is done so because we believe that the trail intensities of such positions fairly close to the position of job i need to be updated in the same way as the position of job i, with the number of such positions being governed by the number of jobs in the given problem (also see the related observations by Rajendran and Ziegler (2004) and (2005) in the case of single-objective flowshop-scheduling problems). We have 2 in the denominator in Expression (3.4) because we use two ants in our MOACA. It is also to be noted that the value of *cut_off* is not static across all iterations (with each iteration involving the generation of two ant sequences) carried out for the given w_1 and w_2 . After the generation of two ant sequences (to be presented in the following), we set *cut_off* equal to (*cut_off*×0.9), and we do the task of updating the pheromone values, as given in Expressions (3.3) and (3.4). In the current study, we initially set *cut_off* to 0.025.

In order to guide the MOACA towards discovering solutions possibly lying on the Pareto-optimal front by making use of S^*, the best sequence obtained so far (for the

given w_1 and w_2, and with respect to Equation (3.1), we supplement the trail intensities as follows.

For $i = 1(1)\ n$ do the following: /*corresponding to n jobs*/
  for $k = 1(1)\ n$ do the following: /*corresponding to n positions*/
    compute $diff = (|\ h^*(i) - k\ | + 1)$
    and
    set $\tau_{ik} := \tau_{ik} + 1\ /\ ((Z(S^*))^p \times (diff)^c)$. (3.5)

In the above, c denotes the power index for *diff*, and $h^*(i)$ refers to the position of job i in sequence S^*. It is evident that τ_{ik} is updated for job i with respect to position k depending upon the relative difference between this position and the position of job i in S^*. Power index c is introduced for enhancing the differentiation.

3.2.4 Construction of Two Ant Sequences and Their Improvement with Respect to the Compromise Objective Function by Local Search Schemes

In the MOACA, a complete sequence is built up, by starting from a null sequence and choosing a job by the following procedure in order to append it to the available partial sequence in position k, for $k = 1, 2, \ldots, n$, and with the initial available partial sequence being a null set.

Set $T_{ik} = \sum_{q=1}^{k} \tau_{iq}$ and sample a uniform random number u in the range (0, 1).

If $u \leq 0.4$
then
  the first job in S^* that is not yet scheduled in the present partial sequence is chosen;
else
  if $u \leq 0.8$
  then
    among the set of the first $(4+\lfloor n/K \rfloor)$ jobs in S^* that are not yet scheduled in the present partial sequence, choose the job with the maximum value of T_{ik};
  else
    job i is selected from the same set of $(4+\lfloor n/K \rfloor)$ unscheduled jobs for position k as a result of sampling from the following probability distribution:

$$p_{ik} = (T_{ik} / \sum_{l} T_{lk}), \tag{3.6}$$

    where job l belongs to the same set of $(4+\lfloor n/K \rfloor)$ unscheduled jobs.

    Note that when there are unscheduled jobs less than this prescribed number, then all such unscheduled jobs are considered for possible selection.

In the above, K is a parameter that helps to decide on the number of unscheduled jobs to be considered while constructing an ant sequence. The rationale behind the selection of the job to be scheduled next is that the choice is governed between the best sequence and the best value of T_{ik} with equal probability, and the probabilistic choice of the job is done with half of the probability of going in for the first unscheduled job found in the best sequence. In addition, the number of unscheduled jobs considered for selection is not the same across all problem sizes, which is quite logical. Readers may see the related works by Rajendran and Ziegler (2004) and (2005) for single-objective flowshop-scheduling problems.

By performing the above procedure for k = 1, 2, ..., n, a complete ant sequence can be generated. Repeat the process for generating one more ant sequence. Check whether an ant sequence can enter the exiting non-dominated front; if so, enter it and accordingly update the front. Note that each of these two sequences is subjected to the JIS and JSS (in different combinations or concatenations) with the consideration of the compromise objective function for the given w_1 and w_2. Let us denote the two final sequences thus obtained by S^1 and S^2 , with the values of the compromise objective functions denoted by $Z(S^1)$ and $Z(S^2)$ respectively. Update Z^* and S^*, if necessary, by comparing Z^* with $Z(S^1)$ and then with $Z(S^2)$. We wish to point out here that we use two different uniform random number streams, while developing an ant sequence: one for sampling u to decide on the choice between the best sequence and the set of unscheduled jobs; and another uniform random number stream for sampling from the probability distribution. Subject S^1 and S^2 to an adjacent pairwise interchange of jobs (interchanging jobs found in positions k and k+1, for k = 1, 2, ..., n-1), thereby generating 2(n-1) sequences in the neighbourhood; and check every generated sequence for possible entry into the non-dominated front and also check for the possible updating of the non-dominated front. Note that these 2(n-1) sequences do not have any impact on trail intensities and that these sequences are generated to primarily explore the neighbourhood for non-dominated solutions.

Set cut_off := (cut_off×0.9), and no_iter := no_iter + 1.

If no_iter is < 16, then repeat the steps given in Sections 3.2.3 and 3.2.4; else increase w_1 by 0.1, decrease w_2 accordingly, initialize cut_off to 0.025 and no_iter to 0, and go back to repeat the steps given in Sections 3.2.2, 3.2.3 and 3.2.4.

It is to be noted that we have opted to vary w_1 from 0 to 1, in steps of 0.1, and set the upper limit on the number of iterations to 16. This is done so in order to restrict the computational effort. Every variant of the proposed MOACA enumerates about $1500n^2$ sequences in the course of the entire search process. It is noteworthy that when each variant has been coded in FORTRAN (DOS version) and executed on a PC with Pentium 4 processor, 3 GHz, 512 MB RAM, a variant requires the execution time of about 10 hours to obtain the non-dominated solutions for all the 90 problem instances. It is also to be noted that the actual execution time is not large in view of the fact that the JIS and JSS are computationally fast schemes, unlike the relatively more-demanding crossover and mutation operations involved in genetic algorithms for permutation flowshop scheduling. Readers may see the related observations by Minella *et* al. (2008) in respect of the MOSA that also uses similar local search schemes involving job insertion or job swap. Of course, one can perform the MOACA

with more enumeration of sequences by increasing the upper limit on the number of iterations so that an enhanced performance of the MOACA can possibly be achieved.

3.3 Step-by-Step Procedure of the MOACA

We now present the step-by-step procedure consolidating the salient features of the MOACA.

Step 1: Set $w_1 = -0.10$ and $w_2 = 1.10$. Obtain an upper bound on makespan and an upper bound on total flowtime for the given problem instance from the available literature, and hence obtain up_C_{max} and up_F.

Step 2: Set $w_1 := w_1 + 0.10$, and $w_2 := w_2 - 0.10$. Generate one seed sequence by ordering the jobs in the ascending order of $\sum_{j=1}^{m}(m-j+1)t_{ij}$ followed by the improvement scheme presented by Nawaz *et al.* (1983) if w_1 is less than or equal to 0.5, or by ordering jobs in the non-increasing order of the sum of process times of jobs, and then using the improvement scheme presented by Nawaz *et al.* if w_1 is greater than 0.5, with the consideration of the current w_1 and w_2. Generate the second seed sequence randomly. Update the non-dominated front by considering these two seed sequences. Every seed sequence is then subjected to the given concatenation of JIS and JSS. Call the final sequences S^1 and S^2. Update S^* and $Z(S^*)$ by using S^1 and S^2. In addition apply adjacent pairwise interchange of jobs to the sequences S^1 and S^2.

Note: Check each sequence generated for non-dominance and if necessary update the non-dominated front.

Initialize τ_{ik} as per Eq. (3.2). Set $cut\_off = 0.025$ and $no\_iter = 0$.

Step 3: Modify τ_{ik} as given by Eq. (3.3), and modify also by reckoning with S^1 and S^2, see Exp. (3.4), and thereafter reckoning with S^*, see Exp. (3.5).

Step 4: Construct two ant sequences by using the procedure given in Section 3.2.4, with each sequence thereafter improved by the given concatenation of JIS and JSS. Call the final sequences S^1 and S^2. Update S^* and $Z(S^*)$ by using S^1 and S^2. In addition apply adjacent pairwise interchange of jobs to the sequences S^1 and S^2.

Note: Check each sequence generated for non-dominance and if necessary update the non-dominated front.

Step 5: Set $cut\_off := cut\_off \times 0.9$ and $no\_iter := no\_iter + 1$. If $no\_iter < 16$ then go to Step 3; else return to Step 2 as long as $w_1 \leq 0.9$. Return the final set of heuristically non-dominated solutions for the given problem instance. STOP.

4 Performance Analysis of the MOACA

In line with previous researchers, we have considered the ninety benchmark flowshop scheduling problem instances by Taillard (1993), with the number of jobs being 20, 50 and 100, and with the number of machines being 5, 10 and 20. In order to evaluate the performance of a multi-objective flowshop scheduling algorithm, many researchers basically used the following metric in one form or another: the number of solutions contributed by an algorithm to the final or net non-dominated front (e.g., Ishibuchi and Murata (1998), Chang *et al.* (2002), Varadharajan and Rajendran

(2005), and Framinan and Leisten (2007)). We have also used a similar metric given as follows:

> number of solutions contributed by a given multi-objective algorithm to the net non-dominated front / total number of solutions in the net non-dominated front. (4.1)

This metric is relatively easy in terms of comprehending how a multi-objective algorithm performs in relation to other multi-objective algorithms.

As the first step, we have set p = 2, 1 and 1.5, with K = 50 and 20, c = 1 and ρ = 0.75 in the proposed MOACA. Note that for every given p, c and ρ, we experiment with K = 50 and 20. The reason is that the parameter K is involved in the generation of an ant sequence, and hence we would prefer to always experiment with these two values of K. The setting of ρ in the neighbourhood of 0.75 is found to work well by previous researchers as well (Stuetzle (1998), and Rajendran and Ziegler (2004) and (2005)). The corresponding variants are termed Variants 1-6 of the MOACA. From the performance analyses of these variants, we have observed that these variants perform not much differently on an average, and every variant does contribute to the final or net non-dominated front in a similar manner. Hence we have decided to freeze p at 1.5. We now set c = 2, and ρ = 0.75 and 0.7. We find that these variants, namely Variants 7-10, do perform well, especially in the case of the larger-sized problems. For the further two variants (namely Variants 11 and 12), we set c = 1.5 and ρ = 0.725. We find that these two variants also perform well, especially for the larger-sized problems. Note that in all these variants, the concatenation of JIS and JSS is done in the following manner: JIS-JSS-JIS-JSS.

As further analysis, we have decided to see the performance of the MOACA by changing the concatenation of the JIS and JSS. First picking up on Variants 11 and 12, we have experimented with the following concatenation of the JIS and JSS: JIS-JIS-JSS-JIS, followed by the concatenation of JIS-JIS-JIS-JSS. The corresponding variants are termed Variants 13 and 14 (derived from Variant 11), and Variants 15 and 16 (derived from Variant 12). Similarly, we have derived Variants 17-20 from Variants 9 and 10 respectively by implementing these two concatenations of JIS and JSS. Our performance analyses have shown that different concatenations of JIS and JSS have indeed served to discover additional non-dominated solutions, especially in the case of larger-sized problems. Table 1 presents the details of settings for different variants of the MOACA.

As mentioned earlier, we have made use of the benchmark solutions provided by Framinan and Leisten (2007), and consolidated the solutions yielded by all the variants of the MOACA with those obtained by Framinan and Leisten. The final sets of non-dominated solutions thus obtained for every problem instance are given in Tables 2a - 4c. We believe that these solutions can possibly serve as benchmarks for future research attempts as much the solutions obtained by Framinan and Leisten served for us as benchmarks. It is be noted again that the solutions obtained by Framinan and Leisten are through the implementation of their MOIGS with eight different values for d, and from the consolidation with the solutions reported by Varadharajan and Rajendran (2005).

We have also evaluated every variant of the MOACA and the set of solutions obtained by Framinan and Leisten (denoted by F&L), by using the metric given in

Table 1. Settings for MOACA variants

MOACA Variant	ρ	*c*	*p*	*K*	Concatenation of local search schemes
1	0.75	1	2	50	JIS-JSS-JIS-JSS
2	0.75	1	2	20	JIS-JSS-JIS-JSS
3	0.75	1	1	50	JIS-JSS-JIS-JSS
4	0.75	1	1	20	JIS-JSS-JIS-JSS
5	0.75	1	1.5	50	JIS-JSS-JIS-JSS
6	0.75	1	1.5	20	JIS-JSS-JIS-JSS
7	0.75	2	1.5	50	JIS-JSS-JIS-JSS
8	0.75	2	1.5	20	JIS-JSS-JIS-JSS
9	0.7	2	1.5	50	JIS-JSS-JIS-JSS
10	0.7	2	1.5	20	JIS-JSS-JIS-JSS
11	0.725	1.5	1.5	50	JIS-JSS-JIS-JSS
12	0.725	1.5	1.5	20	JIS-JSS-JIS-JSS
13	0.725	1.5	1.5	50	JIS-JIS-JSS-JIS
14	0.725	1.5	1.5	20	JIS-JIS-JSS-JIS
15	0.725	1.5	1.5	50	JIS-JIS-JIS-JSS
16	0.725	1.5	1.5	20	JIS-JIS-JIS-JSS
17	0.7	2	1.5	50	JIS-JIS-JSS-JIS
18	0.7	2	1.5	20	JIS-JIS-JSS-JIS
19	0.7	2	1.5	50	JIS-JIS-JIS-JSS
20	0.7	2	1.5	20	JIS-JIS-JIS-JSS

Exp. (4.1). The results of such an analysis are presented in Table 5. In Table 5, an entry in a given row under a given approach denotes the number of non-dominated solutions contributed by a given approach to the net non-dominated front for that problem instance. We then compute the metric given in Exp. (4.1), and sum it up with respect to that approach over ten problem instances. The average over these ten problem instances for that approach is then computed and reported, see the last row in a given problem set or size. Note that for the problem instances with jobs equal to 20,

Table 2a. Net set of non-dominated solutions obtained for the problem size (20×5)

Problem 1		Problem 2		Problem 3		Problem 4		Problem 5	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
1278	14064	1359	16112	1081	13818	1293	16619	1235	14163
1313	14058	1360	16071	1085	13759	1299	16342	1239	14151
1315	14048	1361	15567	1086	13666	1301	15983	1243	14047
1324	14041	1364	15548	1095	13665	1304	15925	1244	14002
1339	14033	1368	15535	1096	13587	1306	15852	1250	13943
		1372	15525	1097	13560	1307	15844	1254	13927
		1377	15454	1099	13524	1309	15828	1264	13890
		1379	15450	1100	13505	1311	15819	1266	13885
		1383	15156	1107	13496	1313	15793	1278	13875
		1385	15151	1111	13418	1319	15758	1285	13872
				1122	13400	1320	15587	1289	13834
				1140	13358	1329	15484	1298	13827
				1183	13347	1354	15447	1301	13811
				1289	13301			1305	13763
								1311	13732
								1328	13668
								1338	13619
								1360	13552
								1387	13529

Table 2a. (*continued*)

Problem 6		Problem 7		Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
1195	14908	1242	14280	1206	14581	1230	14977	1108	13649
1198	14889	1245	14200	1211	14429	1232	14936	1112	13584
1200	14888	1252	13999	1212	14345	1245	14929	1113	13514
1202	14741	1254	13887	1213	14302	1246	14902	1115	13429
1203	14699	1255	13749	1214	14253	1247	14756	1133	13419
1207	14681	1264	13730	1217	14157	1248	14729	1134	13344
1210	14346	1265	13728	1222	14153	1249	14715	1138	13173
1213	14247	1266	13722	1226	14148	1253	14505	1150	13126
1217	14121	1267	13695	1229	14128	1255	14493	1151	13122
1218	14042	1274	13632	1233	14093	1256	14485	1153	13026
1224	13608	1278	13578	1234	14080	1259	14449	1163	12999
1233	13583	1283	13548	1240	14075	1271	14446	1179	12981
1241	13581			1245	14073	1272	14386	1184	12943
1245	13437			1246	14072	1281	14367		
1247	13412			1247	14059	1284	14329		
1248	13410			1252	14051	1336	14317		
1251	13391			1254	13994	1337	14295		
1252	13280			1320	13987				
1255	13274			1329	13948				
1256	13212								
1257	13171								
1260	13160								
1261	13139								
1266	13123								

Table 2b. Net set of non-dominated solutions obtained for the problem size (20×10)

Problem 1		Problem 2		Problem 3		Problem 4		Problem 5	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
1582	22121	1664	23888	1496	20905	1377	19738	1419	19277
1583	21731	1666	23877	1501	20672	1380	19721	1420	19205
1590	21706	1667	23527	1508	20433	1385	19579	1422	19203
1592	21421	1668	23525	1515	20364	1386	19533	1432	18966
1595	21420	1671	23519	1521	20118	1387	19523	1435	18952
1608	21385	1672	23399	1534	20061	1392	19431	1446	18873
1629	21337	1676	23375	1546	20036	1393	19413	1463	18829
1640	21284	1683	23356	1547	20003	1394	19410	1466	18798
1641	21204	1684	23303	1577	19962	1397	19344	1473	18794
1656	21122	1690	23274	1589	19958	1399	19280	1476	18766
1685	21025	1692	23242	1615	19927	1403	19273	1485	18754
1686	21011	1694	23166	1624	19917	1406	19177	1486	18641
1698	21003	1699	23156	1650	19877	1409	19149		
1705	20957	1700	23112	1693	19861	1416	19094		
1707	20911	1701	22999	1703	19833	1424	19082		
		1706	22995			1425	19044		
		1708	22853			1432	19020		
		1728	22807			1437	18992		
		1737	22726			1443	18987		
		1744	22720			1445	18948		
		1764	22714			1451	18908		
		1779	22711			1473	18893		
		1781	22617			1476	18852		
		1782	22608			1493	18828		
		1818	22606			1494	18800		
		1827	22559			1509	18792		
		1831	22524			1525	18751		
		1841	22492			1558	18750		
		1847	22473						
		1872	22446						
		1893	22440						

Table 2b. (*continued*)

Problem 6		Problem 7		Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
1397	20725	1484	19232	1544	22075	1593	20779	1591	22719
1402	20612	1492	19166	1545	21827	1597	20765	1595	22575
1403	20512	1498	19159	1546	20927	1602	20763	1598	22334
1404	20374	1500	18894	1552	20845	1607	20761	1604	21945
1409	20278	1510	18846	1553	20756	1608	20725	1608	21930
1413	20127	1525	18765	1556	20674	1612	20651	1612	21882
1421	20102	1526	18658	1561	20489	1616	20592	1630	21872
1423	20049	1533	18598	1570	20480	1622	20591	1632	21675
1424	20027	1540	18584	1573	20471	1627	20564	1642	21662
1427	20024	1543	18526	1577	20466	1635	20538	1647	21659
1429	19896	1550	18476	1578	20381	1648	20487	1652	21632
1436	19862	1562	18445	1579	20374	1657	20454	1659	21581
1440	19856	1579	18409	1589	20358	1668	20421	1671	21497
1441	19838	1594	18377	1598	20347	1669	20419	1681	21462
1442	19775	1600	18376	1608	20288	1676	20412	1684	21459
1448	19774	1617	18363	1641	20241	1677	20374	1685	21453
1450	19761					1685	20356	1712	21418
1451	19721					1749	20347	1735	21405
1455	19711					1762	20330	1768	21402
1471	19666							1770	21359
1475	19641							1774	21352
1477	19543							1778	21320
1483	19522								
1498	19517								
1522	19495								
1523	19473								
1529	19422								
1551	19382								
1554	19357								
1568	19340								
1594	19331								
1635	19290								
1655	19283								
1659	19249								
1696	19245								

Table 2c. Net set of non-dominated solutions obtained for the problem size (20×20)

Problem 1		Problem 2		Problem 3		Problem 4		Problem 5	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
2297	35831	2099	33261	2328	36809	2223	33282	2294	36054
2298	35764	2100	32912	2332	36578	2224	32841	2300	36040
2299	35724	2104	32874	2336	35985	2225	32546	2305	35992
2300	35665	2105	32786	2353	35829	2233	32516	2309	35834
2301	35623	2111	32769	2363	35821	2234	32231	2314	35608
2302	35384	2118	32762	2366	35739	2249	32124	2322	35528
2303	35358	2119	32734	2369	35363	2251	32121	2336	35451
2310	35322	2120	32684	2373	35251	2253	32025	2337	35440
2313	35274	2125	32681	2383	35243	2260	31993	2343	35365
2317	35237	2129	32647	2385	35217	2261	31928	2345	35215
2324	35195	2132	32489	2388	35120	2263	31855	2390	35214
2325	34965	2145	32482	2395	35094	2264	31826	2399	35154
2341	34961	2147	32462	2399	34991	2265	31804	2401	35131
2344	34954	2149	32360	2400	34959	2276	31753	2402	35076
2345	34738	2153	32339	2402	34917	2289	31726	2411	34942
2346	34581	2154	32316	2407	34840	2296	31714	2434	34805
2351	34533	2163	32205	2414	34783	2301	31708	2508	34782
2352	34467	2166	32089	2422	34732	2311	31690	2519	34710
2355	34374	2196	31906	2426	34707	2387	31677	2538	34667
2363	34220	2206	31826	2429	34703	2405	31661	2560	34659
2380	34139	2214	31777	2430	34679			2564	34649
2386	34126	2254	31716	2433	34614			2570	34645
2388	34026	2259	31713	2435	34480			2571	34616
2391	33998	2261	31612	2449	34400			2607	34605
2392	33901	2275	31597	2453	34388			2613	34602
2412	33827	2334	31587	2456	34385			2617	34590
2418	33799			2465	34377			2622	34557
2427	33742			2466	34364				
2434	33735			2474	34232				
2437	33623			2484	34127				
				2508	34125				
				2526	34110				
				2535	34107				
				2547	34101				
				2549	34084				
				2554	34082				
				2555	34072				
				2557	34055				
				2564	34051				
				2567	34016				
				2578	33977				
				2579	33932				
				2608	33920				

Table 2c. (*continued*)

Problem 6		Problem 7		Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
2230	34231	2276	34517	2200	34792	2237	34532	2180	33462
2232	33853	2278	33756	2202	34758	2243	34516	2183	33264
2234	33652	2282	33438	2205	34712	2248	34363	2191	33240
2239	33557	2292	33436	2209	34612	2253	34360	2196	33125
2242	33407	2299	33425	2210	34555	2258	34338	2202	32937
2252	33395	2305	33390	2212	34129	2260	34183	2229	32824
2253	33383	2307	33353	2221	34123	2281	34178	2231	32805
2257	33351	2320	33325	2222	33931	2289	34138	2238	32764
2258	33330	2324	33295	2224	33882	2292	34133	2242	32731
2260	33160	2334	33282	2234	33843	2297	34077	2245	32654
2263	32876	2335	33276	2237	33744	2308	34065	2246	32583
2270	32853	2336	33253	2238	33661	2310	34062	2249	32497
2281	32810	2340	33221	2242	33640	2319	34046	2250	32477
2284	32778	2343	33211	2243	33420	2320	34031	2270	32423
2292	32758	2350	33206	2257	33267	2336	34029	2287	32383
2304	32752	2353	33184	2266	33107	2337	34015	2308	32375
2307	32714	2356	33178	2273	33068	2343	33959	2309	32331
2318	32707	2359	33139	2284	33045	2356	33900	2329	32310
2320	32693	2368	33107	2294	32990	2360	33847	2338	32299
2324	32656	2407	32987	2297	32975	2372	33805	2339	32292
2334	32655	2415	32970	2299	32943	2379	33772	2345	32269
2358	32652	2453	32951	2311	32921	2418	33734	2365	32262
2359	32650	2466	32922	2312	32909	2419	33729		
2360	32625			2314	32897	2425	33727		
2365	32616			2318	32880	2427	33722		
2369	32604			2323	32865	2428	33641		
2372	32564			2330	32854	2448	33634		
				2331	32814	2455	33625		
				2341	32803	2458	33623		
				2351	32793	2486	33612		
				2360	32775				
				2373	32679				
				2380	32663				
				2391	32642				
				2393	32629				
				2394	32603				
				2396	32552				
				2408	32524				
				2415	32509				
				2433	32506				
				2470	32499				
				2476	32494				
				2478	32485				
				2492	32444				

Table 3a. Net set of non-dominated solutions obtained for the problem size (50×5)

Problem 1		Problem 2		Problem 3		Problem 4		Problem 5	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
2724	67351	2838	76083	2621	65944	2753	72139	2864	70577
2728	67344	2841	76073	2622	65743	2757	70199	2865	70558
2729	67291	2843	75098	2630	65278	2758	70088	2886	70236
2731	67208	2848	69791	2641	65081	2764	70036	2887	70036
2735	65937	2849	69708	2642	65028	2766	69961	2904	69739
2743	65782	2853	69693	2645	64907	2767	69633		
2744	65776	2854	69656	2648	64851	2768	69613		
2745	65752	2857	69522	2660	64817	2775	69586		
2746	65726	2859	69173	2663	64550	2779	69549		
2747	65698	2860	69167	2665	64232	2782	69499		
2752	65218	2861	69088	2667	64108	2785	69490		
2774	65191	2862	69047	2671	64053	2788	69424		
2840	65168	2864	69011	2672	63930	2792	69408		
		2865	68920	2694	63879	2797	69297		
		2867	68894	2698	63861	2800	69256		
		2875	68840	2703	63859	2806	69080		
		2886	68836	2735	63856	2810	69067		
		2889	68811	2776	63838	2856	69000		
		2890	68798			2889	68968		
		2892	68685			2919	68958		
		2896	68683			2930	68864		
		2910	68641			2931	68814		
		2915	68631						
		2916	68622						
		2917	68617						
		2918	68610						
		2929	68599						
		2930	68589						
		2933	68580						
		2934	68575						
		2937	68540						
		2951	68507						
		2954	68491						
		2957	68457						
		2960	68415						
		2967	68413						

Table 3a. (*continued*)

Problem 6		Problem 7		Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
2829	72618	2725	72171	2683	70478	2554	72756	2782	71801
2832	69630	2732	70120	2686	69432	2560	72368	2783	70644
2839	68900	2736	69580	2694	68338	2561	67091	2784	70600
2841	68787	2737	69491	2697	68208	2564	66120	2785	70563
2845	68346	2741	68586	2703	68033	2565	65558	2789	70193
2846	68343	2743	68487	2704	67890	2566	65552	2791	70080
2847	68095	2745	67533	2705	65088	2569	65522	2792	69884
2882	67833	2746	67482	2706	65082	2570	65196	2793	69882
2886	67424	2758	67380	2707	65030	2571	64442	2794	69620
2888	67342	2760	67212	2710	64985	2572	64360	2796	69597
2894	67264	2767	66685	2713	64932	2573	64313	2803	69573
2978	67258	2768	66662	2718	64889	2577	64303	2833	69564
		2785	66545	2719	64883	2581	64190	2835	69546
		2811	66543	2727	64851	2584	64143	2836	69536
		2936	66508	2748	64835	2589	64122	2839	69516
				2810	64828	2590	64100	2841	69515
				2851	64804	2595	64072	2843	69508
						2596	63998	2844	69489
						2603	63966		
						2610	63929		
						2615	63862		
						2627	63854		
						2628	63846		
						2648	63788		
						2652	63721		
						2654	63686		
						2657	63627		
						2664	63561		
						2697	63559		
						2699	63382		
						2723	63371		
						2739	63350		

Table 3b. Net set of non-dominated solutions obtained for the problem size (50×10)

Problem 1		Problem 2		Problem 3		Problem 4		Problem 5	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
3027	97529	2911	88604	2878	86459	3064	93360	3012	92036
3031	97447	2917	88575	2879	86207	3065	92283	3013	92013
3033	96868	2918	88213	2883	85762	3067	91415	3016	91982
3034	93859	2921	88028	2885	85740	3072	91243	3018	91375
3037	93457	2923	87720	2887	85112	3073	91236	3019	91112
3039	93395	2925	87170	2891	85075	3077	91104	3024	91040
3040	92537	2926	86816	2896	85050	3078	90722	3037	90983
3043	92513	2927	86753	2904	85027	3086	90696	3038	89182
3045	92332	2928	86603	2915	84984	3087	90582	3042	88694
3051	91407	2929	86399	2916	84722	3089	90476	3045	88455
3057	90587	2931	85928	2917	83316	3090	90139	3061	88131
3059	90415	2940	85913	2956	83034	3092	90046	3063	88124
3062	90363	2947	85904	2960	82822	3109	89960	3065	88044
3063	90171	2948	85781	2972	82487	3110	89915	3080	88025
3065	89894	2949	85716	2977	82399	3111	89756	3084	88010
3069	89312	2950	85285	2979	82154	3114	89403	3085	87668
3089	89273	2952	85240	2983	81943	3115	89260	3107	87667
3111	89060	2953	85189	2986	81900	3116	89053	3117	87638
3124	89045	2958	85140	2994	81897	3123	89005	3138	87616
3127	88853	2971	85087	2995	81889	3127	88736	3148	87606
3129	88570	2975	85017	2997	81838	3128	88720	3188	87583
3130	88523	2976	84916	3001	81742	3130	88483	3193	87510
3173	88467	2989	84885	3005	81545	3139	88434	3201	87462
3177	88461	2993	84842	3012	81458	3143	88209		
3202	88435	2994	84839	3013	81234	3150	88083		
3206	88387	2995	84835	3024	81231	3168	88066		
3215	88386	2996	84803	3028	80888	3216	88064		
3265	88299	3005	84673	3105	80828	3218	88028		
3273	88297	3015	84509			3239	87963		
		3020	84469			3262	87892		
		3034	84332			3264	87692		
		3059	84284			3274	87574		
		3083	84262			3287	87509		
		3085	84206			3289	87321		
		3090	84198						
		3095	84099						
		3100	84084						
		3106	83844						
		3108	83812						
		3116	83808						
		3136	83722						

Table 3b. (*continued*)

Problem 6		Problem 7		Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
3043	91681	3115	99295	3043	99269	2908	91310	3112	95762
3044	91470	3124	97762	3045	98936	2909	91160	3113	95376
3045	91268	3126	96113	3046	98444	2910	89958	3118	93950
3047	90923	3127	93536	3048	97405	2920	89740	3121	93676
3056	90902	3128	93535	3050	97236	2923	89384	3129	93632
3057	90901	3129	92846	3052	97222	2949	89152	3131	92599
3060	90720	3131	92808	3055	96960	2952	88649	3138	92289
3064	90563	3133	92305	3056	95518	2972	88346	3139	91716
3065	89754	3136	92229	3057	94768	2988	88278	3142	91666
3075	89376	3138	91984	3058	92976	2994	88126	3146	91342
3076	89361	3140	91586	3061	92738	2996	88095	3147	91275
3077	89335	3157	91198	3064	92537	3002	88018	3149	91256
3080	89174	3158	91167	3066	92496	3017	87488	3150	91148
3082	88786	3161	91150	3067	91428	3025	87269	3152	90902
3084	88749	3165	91082	3069	91420	3026	87250	3157	90839
3087	88704	3169	90859	3072	91389	3031	87063	3158	90081
3099	88698	3170	90814	3074	91327	3044	87031	3164	89992
3111	88679	3176	90802	3077	89908	3073	86984	3192	89946
3114	88603	3180	90770	3078	89746	3077	86977	3198	89804
3116	88579	3182	90678	3082	89709	3078	86974	3204	89709
3119	88552	3191	90509	3083	89595	3079	86894	3208	89535
3120	88128	3196	90485	3086	89553	3080	86866	3241	89231
3167	88113	3197	90446	3087	89541	3090	86862	3261	89209
3172	88066	3201	90402	3091	89504	3100	86631	3270	89082
3179	88019	3202	90391	3092	89474			3272	89075
3183	87996	3207	90349	3093	89416			3275	89054
3205	87873	3213	90337	3101	89336			3276	89051
3244	87850	3221	90300	3113	89322			3279	89033
3340	87826	3229	90261	3118	89316			3335	89019
		3230	90229	3128	89315			3449	88982
		3231	90196	3132	89169				
		3233	90132	3134	88906				
		3266	90095	3139	88863				
		3275	90067	3141	88790				
		3301	90046	3144	88742				
		3328	90042	3146	88737				
		3352	89989	3148	88673				
		3391	89929	3157	88608				
				3169	88593				
				3176	88585				
				3177	88527				
				3178	88334				
				3274	88237				
				3276	88224				
				3319	88185				
				3326	87993				

Table 3c. Net set of non-dominated solutions obtained for the problem size (50×20)

Problem 1		Problem 2		Problem 3		Problem 4		Problem 5	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
3908	137314	3769	125037	3718	121229	3785	127750	3668	125806
3911	136711	3775	124846	3719	121194	3786	127731	3672	125107
3912	130906	3783	124831	3720	118930	3793	127710	3675	124900
3915	130783	3798	124094	3760	118911	3794	127700	3682	124472
3932	129913	3799	123802	3761	118906	3795	127600	3684	124444
3933	129685	3810	123778	3763	118893	3797	126628	3694	122330
3934	129376	3816	123464	3773	118850	3799	126470	3701	122102
3941	129338	3829	123201	3788	118817	3800	125246	3708	122094
3955	129170	3831	123103	3794	118769	3807	124900	3729	122082
3963	129169	3833	123078	3796	118739	3811	124869	3736	121771
3966	128600	3847	122626	3799	118647	3812	124863	3754	121673
3970	128544	3852	122559	3804	118593	3823	124146	3762	121393
3982	128483	3853	122208	3809	118525	3824	123909	3764	121339
3983	128371	3862	121942	3831	118517	3825	123838	3777	121152
3988	128362	3888	121916	3834	118464	3826	123718	3789	121066
3991	128357	3894	121903	3845	118269	3831	123648	3802	120956
4013	128306	3895	121898	3937	118086	3837	123640	3808	120898
4015	128219	3915	121895	3946	118083	3839	123568	3812	120857
4021	127837	3920	121881	4004	118036	3842	123315	3821	120773
4036	127770	3921	121345	4005	117926	3846	123314	3830	120723
4043	127661	3961	121169	4020	117636	3847	123286	3832	120545
4045	127655	3962	121050	4068	117619	3850	122646	3837	120516
4064	127603	3964	120562	4072	117600	3860	122611	3842	120342
4087	127518	4099	120486	4096	117556	3863	122583	3850	120329
4094	127338			4106	117500	3867	122390	3856	120256
4109	127308					3868	122389	3861	120213
4140	127302					3876	122306	3866	120199
4146	127040					3914	122297	3869	119810
4170	127037					3915	122152	3882	119775
4177	126861					3919	122103	3896	119628
4184	126846							3907	119502
4202	126713							3909	119348
								3917	119314
								3931	119313
								3954	119222
								3960	119183
								3969	119165
								3989	119156

Table 3c. (*continued*)

Problem 6		Problem 7		Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
3751	126155	3765	129180	3775	131886	3812	130514	3820	128480
3755	125986	3771	128924	3780	131831	3813	129684	3828	128410
3758	125764	3773	128327	3781	129295	3817	129331	3835	128407
3763	125663	3774	128285	3786	127910	3818	129020	3839	128345
3765	125601	3787	128235	3791	127519	3820	128129	3840	127939
3769	125590	3796	127847	3792	127448	3827	128126	3842	127933
3771	125569	3797	127466	3793	126788	3828	126339	3846	127769
3783	125558	3800	127450	3794	126593	3831	126321	3849	127624
3803	125379	3818	127410	3800	126546	3833	126188	3851	127047
3811	124897	3819	127388	3805	126515	3842	125873	3853	126829
3820	124219	3822	126995	3806	126469	3846	125577	3859	126820
3823	124052	3830	126474	3813	125832	3850	125575	3861	126796
3827	123906	3831	126472	3817	125813	3857	125573	3863	126772
3835	123822	3832	126419	3829	125782	3860	125552	3889	126617
3836	123818	3838	126397	3830	125760	3869	125522	3899	126139
3855	123791	3842	126032	3831	125421	3887	125484	3907	126124
3856	123423	3845	125914	3847	125397	3889	125429	3935	126103
3861	123339	3850	125905	3851	125381	3898	125026	3942	125775
3866	123325	3855	125841	3852	125289	3910	124723	3956	125712
3873	123274	3858	125715	3860	125282	3911	124359	4239	125702
3880	123034	3892	125700	3863	125259	3912	124324	4278	125542
3885	123001	3899	125553	3864	125222	3914	124200		
3889	122989	3901	125551	3865	125189	3921	124185		
3891	122671	3911	125550	3868	125094	3922	124180		
3904	122157	3912	125371	3884	124518	3929	124141		
3915	122114	3914	125366	3889	124512	3932	123818		
3921	122054	3924	125146	3910	124470	3934	123812		
4092	122032	3939	125101	3924	124453	3935	123809		
4098	122012	3948	125090	3944	124449	3936	123356		
4106	121895	3950	125058	3946	124445	3944	123292		
		3965	125043	3949	124381	3979	123081		
		3978	125033	3951	124369	3992	122879		
		3994	124972	3959	124322	3998	122779		
		4002	124959	3968	124199				
		4039	124937	3983	124141				
		4049	124894	3997	124088				
		4051	124816	4044	124072				
		4079	124725	4050	124033				
		4227	124706	4063	124019				
				4259	124007				
				4317	123883				

Table 4a. Net set of non-dominated solutions obtained for the problem size (100×5)

Problem 1		Problem 2		Problem 3		Problem 4		Problem 5	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
5493	262040	5284	247380	5175	271196	5017	269625	5250	255617
5495	257719	5286	247261	5177	268438	5018	264631	5251	255583
5500	256992	5287	247127	5183	267517	5019	244402	5252	255567
5527	256068	5288	247056	5186	267344	5021	234041	5255	247072
5564	256056	5291	246957	5193	255989	5032	232875	5256	246971
5570	256010	5297	246937	5195	253303	5035	230917	5257	246765
5609	255943	5299	246879	5206	244864	5042	230812	5259	246663
		5301	246864	5208	243806	5043	230707	5260	245660
		5302	246245	5209	243748	5044	229982	5261	244612
		5311	245621	5212	242690	5082	229933	5263	244514
		5341	245585	5221	242187	5112	229881	5264	244450
		5345	245578	5239	241938	5182	229866	5267	244135
				5240	241564	5189	229857	5272	244104
				5244	240708	5195	229769	5276	244011
				5250	240634			5298	243919
				5251	240594			5303	243668
				5262	240509			5304	243475
				5267	240412			5305	243376
				5294	240378			5307	243301
				5297	240363			5320	243258
				5368	240282			5324	243155
				5369	240198			5333	242928
								5339	242822

Table 4a. (*continued*)

Problem 6		Problem 7		Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
5135	262520	5247	252470	5094	251687	5448	280016	5322	278222
5139	242167	5251	251231	5096	251658	5454	256252	5328	257564
5141	242024	5255	247623	5097	248092	5465	252622	5329	257174
5143	241709	5256	247599	5100	247674	5469	252421	5330	255535
5146	240991	5257	247270	5101	246505	5470	252258	5334	250817
5148	240787	5265	247167	5102	245524	5471	252173	5342	247238
5150	238910	5270	245856	5104	245488	5474	251859	5346	246752
5156	237401	5275	245558	5105	243726	5477	251854	5348	246746
5157	236947	5276	244786	5106	243411	5479	251584	5372	246241
5158	236466	5277	244483	5108	241799	5481	251560	5386	245651
5159	236098	5279	244321	5111	241677	5513	251541	5389	245545
5161	236039	5282	243988	5121	240670	5519	251448		
5162	236030	5296	243522	5127	240547	5523	251287		
5164	235841	5298	243281	5130	236569	5542	251277		
5172	235832	5305	242469	5133	236238				
5178	235607	5376	242417	5135	235532				
5179	235466			5140	235146				
5181	235425			5150	234557				
5183	235373			5152	234189				
5204	235347			5155	233910				
5220	235326			5159	233908				
5288	235270			5196	233754				
				5218	233738				

most algorithms are able to discover many of the non-dominated solutions on the net front. This is evident from comparing the elements in a given row with the number in the last column of the corresponding row. However, this is not the case when the number of jobs are equal to 50 and 100. It shows that as the number of jobs increases, it becomes increasingly difficult for any single algorithm to discover many non-dominated solutions. In addition, as stated earlier, the different concatenations of the JIS and JSS have served to discover many non-dominated solutions, especially in the case of relatively large-sized problems.

It appears that the proposed variants are able to discover many non-dominated solutions, especially for the larger-sized problems, as opposed to the benchmark solutions provided by Framinan and Leisten. It is interesting to note that we are not able to identify which variant is the best among all proposed ones. This is so because the permutation flowshop scheduling problems become harder to solve as the number of jobs increases, and it requires the implementation of many variants of the MOACA to discover as many non-dominated solutions possible. These observations point to the fact that it is indeed challenging to develop a single multi-objective flowshop scheduling algorithm that can discover many non-dominated solutions, all or most by itself.

Table 4b. Net set of non-dominated solutions obtained for the problem size (100×10)

Problem 1		Problem 2		Problem 3	
C_{max}	F	C_{max}	F	C_{max}	F
5781	339398	5362	299003	5691	300928
5782	339268	5364	297917	5692	300822
5785	337379	5365	297799	5695	299691
5787	317712	5367	297348	5696	299481
5789	317638	5370	297159	5698	299342
5792	315311	5372	297037	5700	298874
5799	315182	5373	296576	5701	298842
5800	314591	5375	296557	5702	298829
5801	314449	5377	289483	5703	298254
5802	313699	5380	289090	5704	298240
5807	313256	5386	286373	5705	296260
5810	312685	5387	283589	5720	295830
5811	312656	5391	283584	5724	295394
5812	311939	5394	283533	5726	295343
5814	311913	5395	283258	5731	295299
5834	311071	5403	282655	5732	295189
5863	309427	5407	282501	5736	295042
5865	309314	5410	282206	5737	295030
5866	309193	5414	281538	5738	294362
5869	309122	5418	281040	5748	294359
5873	308550	5422	280611	5750	294046
5874	306790	5426	280444	5753	293608
5875	306581	5434	279914	5755	293489
5878	306563	5437	279897		
5880	305547	5447	279259		
5884	305467	5449	278666		
5897	305449	5450	278656		
5902	305341	5452	278650		
5910	305076	5462	278591		
5914	304996	5464	278464		
5915	304846	5465	278418		
5920	304829	5575	278415		
5934	304500	5593	278229		
5935	304305	5648	278189		
6010	304148	5653	278142		
6022	304128	5661	278077		

Table 4b. (*continued*)

Problem 4				Problem 5			
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
5826	342759	6048	307189	5501	326804	5649	290309
5828	342586	6053	307102	5505	326777	5650	290256
5829	340868	6063	307047	5507	301808	5670	290138
5831	340587	6065	306547	5509	301785	5672	290086
5837	339158	6069	306527	5512	298541	5673	289995
5839	323104	6074	306430	5520	297036	5676	289959
5852	322128	6086	306249	5521	296912	5769	289957
5855	322122	6088	306165	5530	296881	5797	289915
5860	321668	6105	306153	5535	296620	5799	289827
5861	320315	6153	306137	5536	295339	5821	289714
5865	319834	6157	306034	5537	295293	5831	289682
5868	319585			5545	294866	5835	289667
5870	319446			5548	294856	5836	289589
5871	318276			5549	294668	5869	289498
5872	317697			5552	294610		
5874	317065			5554	293697		
5875	316522			5566	293475		
5880	316507			5569	293399		
5883	313202			5570	293396		
5884	311505			5571	293255		
5886	311480			5573	293120		
5891	310851			5576	293013		
5902	309761			5580	292715		
5907	309571			5584	292566		
5911	309244			5587	292561		
5922	309212			5588	292507		
5923	309156			5589	292506		
5933	309090			5595	292424		
5938	308370			5597	292346		
5943	308319			5600	292268		
5952	308301			5602	292165		
5957	308212			5605	292005		
5987	307969			5607	291647		
5994	307854			5621	291297		
6010	307675			5628	291114		
6037	307488			5631	290582		
6042	307235			5645	290567		
6047	307233			5646	290424		

Table 4b. (*continued*)

Problem 6		Problem 7		Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
5308	310735	5602	299370	5653	313225	5916	354188	5881	328588
5311	310461	5603	298022	5655	312447	5918	353627	5883	328000
5314	309833	5604	298018	5657	308086	5919	349544	5889	327950
5315	309763	5611	296590	5658	307545	5920	348927	5892	327706
5316	309576	5617	296255	5668	306254	5923	347760	5897	327126
5317	292932	5619	296149	5670	306097	5928	319642	5903	300553
5318	292018	5620	295646	5672	306018	5932	319589	5904	300541
5319	291083	5622	292096	5675	304991	5935	319029	5907	300520
5321	289478	5623	292018	5677	303990	5940	313635	5909	300223
5323	282469	5641	290126	5687	303534	5941	312810	5910	299016
5326	281203	5643	288820	5689	303284	5942	312755	5911	298938
5331	280751	5662	288389	5690	302373	5955	312314	5912	298285
5332	280337	5673	287045	5694	301378	5958	312309	5914	298114
5334	280324	5676	286988	5695	299976	5966	311647	5916	298045
5345	279544	5696	286907	5697	299968	5969	311282	5918	297909
5347	279248	5701	285714	5698	299909	5971	310409	5921	297824
5348	278753	5702	284972	5699	299598	5975	310374	5932	297333
5350	278714	5794	284844	5709	299062	5979	310081	5933	297131
5358	278438	5832	284811	5717	298993	5982	310012	5938	296920
5359	278282	5839	284795	5721	298957	5988	309728	5951	296736
5361	277590	5902	284792	5723	298119	5989	309149	5961	296729
5404	277316	5931	284765	5726	298072	6010	308630	5962	296575
5405	277299	5933	284700	5730	298027	6025	308627	5978	296442
5406	277240	5945	284678	5732	297881	6027	308614	6001	296406
5413	277205	5959	284671	5745	297642	6028	308121	6002	296398
5414	277084			5753	297475	6032	308099	6015	296343
5422	276743			5761	296902	6035	307375	6023	296140
5425	276612			5770	296787	6045	307314		
5429	276591			5777	296511	6065	307269		
5432	276275			5825	296203	6098	307260		
5434	276227			5864	296191	6122	307083		
5435	276196			5871	296094	6144	307034		
5440	275644			5881	296083	6163	306982		
5449	275639			5896	296031	6178	306794		
5452	275164			5915	295714				
5454	274997			5988	295638				
5462	274813								
5465	274749								
5510	274701								
5611	274700								

Table 4c. Net set of non-dominated solutions obtained for the problem size (100×20)

Problem 1				Problem 2		Problem 3			
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
6350	394882	6521	375831	6306	396288	6383	400928	6682	379809
6361	391134	6571	375818	6307	395091	6389	400907	6710	379412
6370	390878	6581	375537	6309	394851	6390	400820	6711	379373
6373	390861	6585	375381	6312	394722	6391	400743	6821	379168
6374	390802	6588	375000	6314	394441	6394	400720	6822	379119
6375	390755	6590	374806	6315	393407	6395	392250	6823	379111
6377	389788	6593	374671	6317	393346	6425	391946	6830	379110
6380	389685	6651	374663	6328	393284	6426	387453	6832	379103
6386	386479	6736	374010	6332	391493	6452	386915	6833	379079
6387	386441	6786	373687	6338	391188	6471	386879	6838	379026
6389	386403	6790	373563	6357	391020	6475	386455	6842	378955
6392	386402	6799	373534	6367	390464	6477	384609	6868	378783
6395	386283	6800	373462	6382	390404	6489	384063		
6402	385681	6809	373250	6383	390189	6503	384058		
6403	385482	6814	373218	6385	390149	6504	384040		
6405	385028	6838	373193	6386	390119	6506	383913		
6415	384986	6840	373148	6397	389714	6513	383880		
6418	384977	6902	373140	6398	389576	6514	383868		
6420	384015	6915	373051	6403	387929	6515	383813		
6423	383628			6405	387610	6522	383017		
6424	383613			6408	387375	6523	382953		
6426	383595			6411	387010	6525	382821		
6440	383142			6412	386878	6529	382789		
6444	382900			6413	386726	6530	382774		
6446	382899			6420	385568	6536	382178		
6447	382390			6423	384944	6546	382078		
6451	381468			6426	384892	6547	381796		
6452	381426			6429	383777	6548	381783		
6455	381416			6431	383224	6567	381432		
6457	381372			6433	383211	6575	381220		
6464	381365			6538	382977	6579	381125		
6475	380330			6543	382973	6580	381074		
6476	380215			6549	382925	6581	381052		
6481	380196			6557	382658	6582	380687		
6487	379899			6562	382625	6599	380225		
6492	379866			6572	382540	6600	380222		
6493	379847			6594	382249	6601	380152		
6497	379795			6599	381994	6610	380146		
6506	378838			6625	380997	6651	380101		
6510	378524			6816	380868	6653	379948		
6514	375922			6818	380840	6666	379813		
6520	375833			6835	380664	6667	379812		

Table 4c. (*continued*)

Problem 4				Problem 5		Problem 6	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
6363	403871	6690	382236	6433	394215	6488	394212
6364	401043	6692	382176	6439	388296	6490	394184
6369	399804	6699	381909	6442	387483	6491	394131
6370	399801	6711	381861	6443	386821	6495	394128
6372	399748	6716	381532	6445	386813	6501	394056
6375	398295	6727	381290	6447	386485	6506	392865
6377	398225	6820	381012	6448	386464	6508	391704
6380	397772	6823	380950	6450	386448	6513	391193
6383	397764	6825	380885	6451	386210	6514	391152
6385	396809	6924	380771	6452	386194	6533	389728
6388	396808	6928	380594	6458	385961	6536	389501
6392	396788			6461	385900	6547	389401
6393	396571			6468	385883	6555	388696
6395	396169			6474	384655	6557	387309
6398	395924			6475	384458	6558	386700
6400	395858			6488	384267	6563	386689
6405	394698			6489	384114	6564	386365
6407	392664			6491	383889	6569	386332
6409	392493			6499	383632	6583	385920
6414	392489			6504	381632	6587	385885
6417	392201			6510	381470	6591	385066
6423	390799			6516	381031	6592	384814
6447	389527			6519	381000	6593	384692
6450	388340			6538	380632	6602	384684
6456	387805			6546	380283	6618	383656
6462	387791			6565	380246	6625	383452
6464	387637			6571	380112	6675	383147
6496	387134			6572	379488	6679	383093
6503	387084			6601	379214	6680	383014
6512	386838			6602	378929	6681	382480
6516	386682			6615	378847	6693	382430
6526	386593			6619	378771	6695	382385
6530	386104			6621	378717	6699	382189
6538	386012			6631	378533	6705	381665
6541	385885			6639	378509	6722	381548
6543	385628			6642	377717	6731	381444
6545	385608			6643	377579	6757	380832
6548	383630			6656	377482	6786	380640
6571	383263			6658	377466	6789	380601
6573	383056			6667	377336	6790	380479
6582	383033			6670	377302	6804	380466
6616	382980			6673	377257	6816	380253
6619	382857			6682	376806	6827	380235
6644	382855			6699	376432	6829	379890
6660	382854			6701	376421	6864	379836
6682	382414			6709	375943		
6686	382393			6732	375857		
6688	382306			6733	375346		

Table 4c. (*continued*)

Problem 7				Problem 8		Problem 9		Problem 10	
C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F	C_{max}	F
6394	403384	6654	383872	6541	411389	6413	397716	6528	409325
6395	403352	6658	383581	6543	411309	6415	397701	6529	409286
6397	398870	6663	383551	6549	411152	6419	397305	6544	407571
6398	398859	6665	383232	6557	410992	6425	396813	6545	401166
6400	398775	6669	383185	6566	410727	6426	388701	6551	400822
6419	398477	6677	382680	6567	410658	6430	388686	6556	400049
6421	398436	6679	382665	6568	409888	6433	388652	6559	399751
6431	398154	6688	382360	6570	409723	6435	388620	6564	398717
6432	396999	6698	381782	6573	405352	6451	388522	6569	398087
6435	396899	6699	381671	6575	405308	6456	388273	6570	396745
6442	396896	6701	381579	6579	405213	6468	387912	6573	396017
6445	396365	6824	381504	6585	402822	6470	387370	6583	395563
6450	393029			6592	402554	6479	387006	6585	395375
6456	392973			6599	401899	6480	386888	6588	394639
6458	392923			6600	400210	6483	386846	6603	393528
6461	392831			6614	400202	6494	386821	6606	393255
6466	392755			6615	400093	6497	386799	6624	392822
6470	392704			6638	400061	6507	386701	6625	392802
6471	392650			6639	399929	6523	386308	6626	392699
6473	391149			6641	399572	6532	385566	6628	392668
6477	391045			6643	398987	6533	385554	6637	391442
6478	390977			6644	398953	6540	385510	6644	391163
6485	390884			6652	398752	6543	385057	6654	390946
6490	390566			6653	398713	6545	384624	6663	389961
6499	389357			6654	398649	6568	383717	6667	389957
6509	389329			6662	398398	6571	383642	6672	389951
6510	388908			6677	398346	6689	383329	6674	389815
6512	388044			6678	397274	6695	383090	6678	389796
6513	387837			6679	396833	6701	382870	6688	389734
6515	387810			6680	396752	6774	382841	6693	389711
6516	387804			6694	396682	6786	382622	6708	389661
6549	387666			6706	395013	6795	382492	6714	389348
6554	385481			6711	394922	6803	382326	6721	388859
6563	385460			6721	394850	6812	382320	6725	388567
6577	385174			6731	394669	6834	382269	6734	388468
6578	385009			6733	393458	6863	382215	6742	388048
6581	384358			6847	393310	6871	382200	6764	387826
6627	384326			6849	393077	6874	382151	6769	387817
6628	384311			6874	392935	6907	382089	6813	387803
6635	384299			6891	392906	6919	381969	6845	387564
6636	384046			6982	392592	6944	381918	6852	387169
6639	384008			7018	392477			6858	387096

Table 5. Contributions of the MOACA variants and the solutions from Framinan and Leisten to the net non-dominated front

(*n*×*m*) (20×5)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
1	4	3	4	3	4	3	3	2	3	1	2	4	4	4	4	4	4	4	4	4	1	5
2	3	2	3	0	3	2	0	2	2	1	2	2	0	0	0	1	0	0	0	1	6	10
3	4	6	4	5	4	5	3	3	2	2	2	5	0	1	2	4	0	0	0	4	6	14
4	7	4	7	4	7	4	5	3	6	2	7	2	8	5	7	7	7	5	7	6	0	13
5	12	12	12	12	12	12	13	12	13	12	13	13	14	13	14	13	12	14	13	13	13	19
6	10	8	10	8	10	8	9	9	9	7	10	12	9	8	8	8	9	8	8	7	8	24
7	0	0	0	0	0	0	1	3	1	5	1	0	0	0	0	0	0	4	0	0	2	12
8	9	9	9	9	9	9	6	7	7	7	7	8	4	7	5	5	6	7	5	5	2	19
9	2	11	2	11	2	11	5	5	5	7	5	7	5	10	4	8	4	8	5	4	6	17
10	4	5	4	5	4	5	5	5	4	3	5	8	8	6	4	5	7	3	5	3	2	13
Average	0.39	0.40	0.39	0.37	0.39	0.39	0.33	0.33	0.35	0.30	0.35	0.41	0.37	0.37	0.34	0.39	0.34	0.37	0.33	0.34	0.30	

Absolute numbers, except Average; Average is relative contribution considering all problems in the respective set

Table 5. (*continued*)

(*n*×*m*) (20×10)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
1	8	4	8	6	8	6	2	4	4	4	3	4	6	4	4	5	3	3	2	6	6	15
2	13	3	13	3	13	3	10	9	10	11	9	8	10	12	6	12	11	5	9	12	20	31
3	7	4	8	4	8	4	3	2	1	6	6	3	5	3	6	8	7	5	1	2	6	15
4	8	8	8	8	8	8	8	11	4	4	11	4	9	11	7	6	5	7	7	9	2	28
5	5	2	5	2	5	2	5	5	2	3	1	2	0	1	4	3	0	2	2	2	1	12
6	4	7	4	7	4	7	5	7	7	6	7	8	14	1	5	9	10	3	4	6	0	35
7	7	9	7	9	7	9	5	8	5	8	5	10	12	13	13	15	13	14	13	14	12	16
8	10	2	10	2	10	2	5	1	4	0	5	0	2	5	1	0	3	3	0	1	2	16
9	9	12	8	13	9	12	9	7	11	12	11	10	2	7	5	6	4	6	8	2	8	19
10	5	5	5	5	5	5	5	4	5	8	8	10	5	3	10	3	4	5	4	1	5	22
Average	0.40	0.28	0.40	0.30	0.41	0.30	0.28	0.28	0.25	0.31	0.31	0.29	0.30	0.30	0.32	0.34	0.29	0.28	0.24	0.27	0.31	

Table 5. (*continued*)

(*n*×*m*) (20×20)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**		
1	10	8	7	8	7	8	10	9	6	8	14	9	8	12	11	11	6	8	12	7	23	30
2	4	6	4	6	4	6	4	1	4	2	7	2	3	7	3	7	1	3	3	4	9	26
3	20	20	20	20	20	20	13	14	12	13	20	21	14	16	6	11	11	9	7	15	23	43
4	10	6	10	6	10	6	10	10	10	6	10	8	4	6	7	8	7	8	4	8	6	20
5	7	2	9	2	7	2	2	1	8	0	5	1	7	1	5	1	0	6	2	4	10	27
6	7	9	7	9	7	9	4	8	6	10	4	7	6	6	8	9	6	7	4	8	6	27
7	14	7	14	7	14	7	13	7	10	9	13	11	8	14	5	16	9	13	5	10	17	23
8	12	12	6	12	6	12	3	13	0	6	2	10	10	5	8	5	8	5	9	11	10	44
9	8	6	8	6	8	6	5	10	5	12	10	4	9	10	9	12	7	10	3	10	2	30
10	6	7	6	10	6	7	4	9	5	10	3	10	8	10	11	5	6	7	8	11	6	22
Average	0.34	0.28	0.32	0.29	0.32	0.28	0.25	0.28	0.25	0.27	0.31	0.29	0.26	0.31	0.27	0.31	0.22	0.28	0.20	0.31	0.39	

Table 5. (*continued*)

(*n*×*m*) (50×5)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
1	0	0	0	0	0	0	1	3	1	2	1	0	2	0	1	1	1	0	0	0	0	13
2	4	0	5	0	5	0	3	0	7	1	4	0	1	0	0	1	1	0	0	10	0	36
3	1	1	0	1	0	2	2	0	3	3	2	2	0	1	0	0	1	0	2	0	0	18
4	0	4	0	0	0	4	2	0	6	1	0	4	3	0	1	0	1	0	0	0	0	22
5	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	1	0	0	5
6	0	0	0	1	0	0	0	2	0	3	0	1	1	1	0	0	0	0	2	1	0	12
7	2	0	2	0	2	0	2	1	0	0	0	1	0	1	0	0	3	2	2	0	1	15
8	0	1	0	1	0	1	1	1	0	1	0	5	0	0	1	0	6	0	0	1	0	17
9	0	3	0	1	0	1	5	0	2	0	3	2	5	0	0	1	3	3	6	0	0	32
10	0	3	0	0	0	2	1	6	0	0	7	1	0	0	0	0	0	0	0	0	0	18
Average	0.03	0.08	0.03	0.02	0.03	0.05	010	0.11	0.08	0.07	0.08	0.09	0.06	0.02	0.02	0.01	0.09	0.04	0.08	0.04	0.01	

Table 5. (*continued*)

(*n*×*m*) (50×10)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
1	3	0	3	1	3	1	4	2	1	0	4	4	2	1	0	2	2	3	0	0	0	29
2	3	0	3	1	3	1	0	0	1	6	4	0	1	7	0	3	7	1	2	5	0	41
3	3	8	3	1	3	8	1	0	1	0	5	2	3	0	2	0	0	1	0	1	0	28
4	0	1	1	1	0	0	0	0	0	2	5	0	0	1	0	8	7	3	0	2	3	34
5	0	0	0	0	0	0	0	0	0	0	2	0	0	5	0	0	0	0	11	5	0	23
6	0	0	0	1	0	1	0	0	0	2	6	8	8	0	0	1	1	0	1	1	0	29
7	0	1	0	2	0	1	7	2	0	4	4	6	0	1	0	3	2	3	3	1	0	38
8	1	0	1	0	4	0	0	0	0	1	1	5	2	12	4	2	2	3	7	2	0	46
9	0	3	6	0	0	0	3	1	0	1	0	0	2	0	3	2	0	0	3	0	0	24
10	0	4	0	3	0	2	0	0	0	5	0	6	3	8	0	0	2	1	0	0	0	30
Average	0.03	0.06	0.06	0.03	0.04	0.05	0.05	0.02	0.01	0.06	0.10	0.10	0.07	0.10	0.03	0.06	0.06	0.04	0.09	0.05	0.01	

Table 5. (*continued*)

(*n*×*m*) (50×20)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
1	2	0	4	2	5	2	0	0	8	1	1	0	3	0	0	2	1	0	3	0	0	32
2	0	4	0	0	0	1	4	0	0	1	0	0	0	0	4	3	0	5	3	0	0	24
3	0	0	1	0	0	0	0	0	4	0	0	0	0	0	5	1	6	0	3	3	2	25
4	1	0	0	0	0	0	0	3	0	10	0	0	0	0	1	0	5	7	0	3	0	30
5	0	0	0	0	0	0	0	2	9	0	0	0	0	0	1	2	9	0	15	0	0	38
6	0	0	0	0	0	0	4	3	0	2	0	0	0	0	2	7	0	1	7	0	4	30
7	0	0	0	0	0	1	3	11	5	0	3	5	1	4	3	0	0	0	3	0	0	39
8	0	0	0	8	0	6	0	8	1	0	1	2	1	5	0	4	0	8	2	1	0	41
9	0	0	0	0	0	0	3	0	0	3	0	0	6	0	6	0	11	2	0	1	1	33
10	0	0	0	0	0	0	0	0	0	1	0	2	1	4	1	0	10	1	1	0	0	21
Average	0.01	0.02	0.02	0.03	0.02	0.03	0.05	0.07	0.08	0.06	0.01	0.03	0.04	0.04	0.08	0.06	0.15	0.08	0.11	0.03	0.02	

Table 5. (*continued*)

(*n*×*m*) (100×5)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
1	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0	4	0	0	0	0	7
2	0	2	0	0	0	0	3	1	1	0	0	0	1	0	0	0	0	0	4	0	0	12
3	2	0	1	0	0	0	1	1	1	7	1	2	3	3	1	0	1	0	0	0	0	22
4	4	0	4	0	4	1	3	0	0	0	1	3	0	0	0	0	0	0	0	0	0	14
5	3	3	2	0	2	0	0	2	0	1	6	1	0	3	0	0	0	0	0	2	0	23
6	0	2	6	0	1	0	0	1	7	1	1	0	0	0	0	2	0	0	1	0	0	22
7	0	2	0	2	0	2	3	2	4	1	2	2	0	1	0	0	1	1	0	0	0	16
8	1	0	7	1	1	1	0	0	0	0	3	5	1	1	0	2	1	1	0	0	0	23
9	2	1	1	0	0	0	0	0	1	1	0	0	0	0	0	4	0	0	3	2	0	14
10	1	2	1	0	0	0	1	0	0	0	2	0	0	1	0	0	2	1	0	0	0	11
Average	0.08	0.08	0.12	0.02	0.05	0.02	0.08	0.04	0.08	0.05	0.12	0.07	0.04	0.05	0.01	0.05	0.09	0.02	0.06	0.02	0.00	

Table 5. (*continued*)

(*n*×*m*) (100×10)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**		
1	2	0	1	2	3	1	3	0	3	8	2	0	0	1	0	0	7	0	4	0	0	36
2	2	0	2	0	0	0	0	2	3	0	3	0	4	1	1	0	10	0	10	0	0	36
3	8	3	0	0	2	0	0	5	0	0	2	1	0	0	0	0	2	0	0	0	0	23
4	1	0	5	3	8	0	2	3	5	8	1	0	0	4	0	1	8	0	4	1	0	49
5	2	0	0	0	3	2	0	0	4	0	5	2	7	2	0	6	0	11	2	6	0	52
6	0	0	3	0	0	0	4	4	3	1	12	0	2	3	0	0	0	0	7	1	0	40
7	0	2	0	5	0	1	4	0	1	0	4	0	1	0	1	1	2	3	2	0	0	25
8	2	1	4	0	3	2	0	0	2	0	0	6	1	6	1	0	0	7	2	3	0	36
9	3	3	2	0	0	0	0	1	0	0	0	0	1	8	3	1	6	2	1	3	0	34
10	0	2	5	0	0	0	6	0	0	2	3	0	4	1	0	2	1	0	1	0	0	27
Average	0.07	0.04	0.06	0.03	0.05	0.02	0.06	0.05	0.05	0.05	0.09	0.03	0.05	0.07	0.02	0.03	0.10	0.06	0.09	0.03	0.00	

Table 5. (*continued*)

(*n*×*m*) (100×20)	MOACA variants																				F&L	Number of solutions in the net front
Problem Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
1	3	0	0	0	0	0	7	0	19	0	0	0	0	0	11	4	5	0	8	3	1	61
2	4	0	0	0	0	0	0	0	7	2	9	1	6	0	2	0	2	0	2	7	0	42
3	2	3	9	0	1	0	5	2	2	7	7	3	0	1	0	0	1	4	7	0	0	54
4	1	0	0	0	3	0	0	1	2	3	0	0	8	4	15	5	8	7	1	0	1	59
5	0	2	3	0	2	0	0	0	0	0	0	9	1	2	0	0	19	2	2	8	0	48
6	3	0	5	0	0	0	0	0	2	0	2	5	7	6	0	0	6	5	4	0	0	45
7	0	2	2	0	0	0	0	0	0	5	2	7	4	3	2	0	8	15	0	4	0	54
8	0	0	1	0	0	3	8	3	7	0	2	0	0	6	0	0	2	0	10	0	0	42
9	6	0	0	0	0	0	4	0	0	0	1	0	0	2	4	0	8	8	0	8	0	41
10	0	1	1	0	0	0	4	0	3	0	0	1	9	2	6	3	2	5	3	2	0	42
Average	0.04	0.02	0.04	0.00	0.01	0.01	0.06	0.01	0.08	0.03	0.05	0.05	0.07	0.06	0.08	0.02	0.13	0.09	0.08	0.07	0.00	

5 Conclusions

The problem of scheduling in permutation flowshops with the objectives of minimizing the makespan and total flowtime of jobs was investigated. A new multi-objective ant-colony algorithm, called MOACA, has been developed with many unique features. Twenty variants of MOACA have been proposed. Benchmark flowshop scheduling problems have been solved by using these variants of the MOACA, and a non-dominated solution front is obtained by consolidating the solutions obtained from these variants and the benchmark solutions available in the literature. It is evident from the computational evaluation that the proposed variants of the MOACA are quite effective in discovering many non-dominated solutions. We believe that the non-dominated solutions obtained by us could serve as possible benchmarks for future researchers as much as we have benefited from the earlier researchers. The complete set of non-dominated solutions for every problem instance is given to serve an as easy reference for future researchers.

Acknowledgments. The first author thanks the Alexander-von-Humboldt Foundation for supporting him through the Fellowship in 2003, 2004 and 2006. Thanks are also due to Varadharajan, Madhushini and Christian Petri for their help in consolidating the results. Special thanks are due to Jose Framinan and Rainer Leisten for sharing their benchmark solutions with us.

Appendix

The step-by-step procedure of the job-index-based insertion scheme (JIS) is presented below.

Step 1: Let the input sequence to the JIS be denoted, in general, by S. Let $Z(S)$ denote its compromise objective function value for the given w_1 and w_2. Let $[k]$ denote the job found in position k of S. Initialize $i = 0$.

Step 2: Set $i := i + 1$.

Step 3: For $k = 1$ to n do the following:

if $i \neq [k]$

then

remove job i from its current position in S, insert job i in position k of S and adjust the sequence accordingly by not changing the relative positions of other jobs in S. Let the resultant sequence be denoted by Σ^k; calculate its makespan and total flowtime denoted respectively by $C_{max}(\Sigma^k)$ and $F(\Sigma^k)$; let its compromise objective function value be denoted by $Z(\Sigma^k)$; check if Σ^k enters the non-dominated front, and if so, accordingly update the front; also, if $C_{max}(\Sigma^k) < up\_C_{max}$, set $up\_C_{max} = C_{max}(\Sigma^k)$; and likewise, if $F(\Sigma^k) < up\_F$, set $up\_F = F(\Sigma^k)$.

else

set $k' = k$.

Step 4: Determine sequence Σ^l such that

$Z(\Sigma^l) = \min\{ Z(\Sigma^k)$ for $k = 1, 2, \ldots, n$, and $k \neq k' \}$.

If $Z(\Sigma^l) < Z(S)$ then set $S = \Sigma^l$ and $Z(S) = Z(\Sigma^l)$.

Step 5: Go back to Step 2 if $i < n$; else stop. Sequence S is the output sequence from the JIS.

The step-by-step procedure of the job-index-based swap scheme (JSS) is presented below.

Step 1: Let the input sequence to the JSS be denoted, in general, by S. Let $Z(S)$ denote its compromise objective function value for the given w_1 and w_2. Let $[k]$ denote the job found in position k of S. Initialize $i = 0$.

Step 2: Set $i := i + 1$.

Step 3: For $k = 1$ to n do the following:

if $i \neq [k]$

then

generate sequence Σ^k which differs from S only by having swapped jobs i and $[k]$; calculate its makespan and total flowtime; let its compromise objective function value be denoted by $Z(\Sigma^k)$; check if Σ^k enters the non-dominated front, and if so, accordingly update the front; also, if $C_{max}(\Sigma^k) < up\_C_{max}$, set $up\_C_{max} = C_{max}(\Sigma^k)$; and likewise, if $F(\Sigma^k) < up\_F$, set $up\_F = F(\Sigma^k)$

else

set $k' = k$.

Step 4: Determine sequence Σ^l such that

$Z(\Sigma^l) = \min\{ Z(\Sigma^k)$ for $k = 1, 2, \ldots, n$, and $k \neq k' \}$.

If $Z(\Sigma^l) < Z(S)$ then set $S = \Sigma^l$ and $Z(S) = Z(\Sigma^l)$.

Step 5: Go back to Step 2 if $i < n$; else stop. Sequence S is the output sequence from the JSS.

References

Allahverdi, A.: A new heuristic for m-machine flowshop scheduling problem with bicriteria of makespan and maximum tardiness. Computers & Operations Research 31, 157–180 (2004)

Allahverdi, A., Aldowaisan, T.: New heuristics to minimize total completion time in m-machine flowshops. International Journal of Production Economics 77, 71–83 (2002)

Armentano, V.A., Arroyo, J.E.C.: An application of a multi-objective tabu search algorithm to a bicriteria flowshop problem. Journal of Heuristics 10, 463–481 (2004)

Arroyo, J.E.C., Armentano, V.A.: Genetic local search for multi-objective flowshop scheduling problems. European Journal of Operational Research 167, 717–738 (2005)

Bagchi, T.P.: Multiobjective scheduling by genetic algorithms. Kluwer Academic Publishers, Boston (1999)

Ben-Daya, M., Al-Fawzan, M.: A tabu search approach for the flow shop scheduling problem. European Journal of Operational Research 109, 88–95 (1998)

Campbell, H.G., Dudek, R.A., Smith, M.L.: A heuristic algorithm for the n-job, m-machine sequencing problem. Management Science 16, B630–B637 (1970)

Chakravarthy, K., Rajendran, C.: A heuristic for scheduling in a flowshop with the bicriteria of makespan and maximum tardiness minimization. Production Planning and Control 10, 707–714 (1999)

Chang, P.-C., Hsieh, J.-C., Lin, S.G.: The development of gradual priority weighting approach for the multi-objective flowshop scheduling prob¬lem. International Journal of Production Economics 79, 171–183 (2002)

Chung, C.-S., Flynn, J., Kirca, O.: A branch and bound algorithm to minimize the total flow time for m-machine permutation flowshop problems. International Journal of Production Economics 79, 185–196 (2002)

Corne, D.W., Knowles, J.D., Oates, M.J.: The pareto envelope-based selection algorithm for multiobjective optimization. In: Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Guervos, J.J.M., Schwefel, H.P. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 839–848. Springer, Heidelberg (2000)

Corne, D.W., Jerram, N.R., Knowles, J.D., Oates, M.J.: PESA-II: Region-based selection in evolutionary multiobjective optimization. In: Spector, L., Goodman, E.D., Wu, A., Langdon, W.B., Voigt, H.M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M.H., Burke, E. (eds.) Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001), pp. 283–290. Morgan Kaufmann, San Francisco (2001)

Daniels, R.L., Chambers, R.J.: Multiobjective flow-shop scheduling. Naval Research Logistics 37, 981–995 (1990)

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6, 182–197 (2002)

Dong, X., Huang, H., Chen, P.: An improved NEH-based heuristic for the permutation flowshop problem. Computers & Operations Research 35, 3962–3968 (2008)

Dorigo, M.: Optimization, Learning and Natural Algorithms (in Italian). PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy (1992)

Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man and Cybernetics – Part B 26, 29–41 (1996)

Framinan, J.M., Leisten, R.: An efficient constructive heuristic for flowtime minimisation in permutation flow shops. OMEGA 31, 311–317 (2003)

Framinan, J.M., Leisten, R.: A heuristic for scheduling a permutation flowshop with makespan objective subject to maximum tardiness. International Journal of Production Economics 99, 28–40 (2006)

Framinan, J.M., Leisten, R.: A multi-objective iterated greedy search for flowshop scheduling with makespan and flowtime criteria. OR Spectrum, published online before print, August 4 (2007)

Framinan, J.M., Leisten, R., Ruiz-Usano, R.: Efficient heuristics for flowshop sequencing with the objectives of makespan and flowtime minimisation. European Journal of Operational Research 141, 559–569 (2002)

Framinan, J.M., Ruiz-Usano, R., Leisten, R.: Comparison of heuristics for flowtime minimisation in permutation flowshops. Computers & Operations Research 32, 1237–1254 (2005)

Garey, M.R., Johnson, D.S., Sethi, R.: The complexity of flowshop and jobshop scheduling. Mathematics of Operations Research 1, 117–129 (1976)

Geiger, M.J.: On operators and search space topology in multi-objective flow shop scheduling. European Journal of Operational Research 181, 195–206 (2007)

Gelders, L.F., Sambandam, N.: Four simple heuristics for scheduling a flow-shop. International Journal of Production Research 16, 221–231 (1978)

Ho, J.C.: Flowshop sequencing with mean flow time objective. European Journal of Operational Research 81, 571–578 (1995)

Ignall, E., Schrage, L.: Application of the branch-and-bound technique to some flowshop scheduling problems. Operations Research 13, 400–412 (1965)

Ishibuchi, H., Murata, T.: A multi-objective genetic local search algorithm and its application to flowshop scheduling. IEEE Transactions on Sys¬tems, Man and Cybernetics – Part C: Applications and Reviews 28, 392–403 (1998)

Johnson, S.M.: Optimal two- and three-stage production schedules with setup times included. Naval Research Logistics Quarterly 1, 61–68 (1954)

Kalczynski, P.J., Kamburowski, J.: On the NEH heuristic for minimizing the makespan in permutation flow shops. OMEGA 35, 53–60 (2007)

Kalczynski, P.J., Kamburowski, J.: An improved NEH heuristic to minimize makespan in permutation flow shops. Computers & Operations Research 35, 3001–3008 (2008)

Laha, D., Chakraborty, U.K.: A constructive heuristic for minimizing makespan in no-wait flow shop scheduling. International Journal of Advanced Manufacturing Technology (2008) (DOI: 10.1007/s00170-008-1454-0)

Liao, C.-J., Tseng, C.-T., Luarn, P.: A discrete version of particle swarm optimization for flowshop scheduling problems. Computers & Operations Research 34, 3099–3111 (2007)

Liu, J., Reeves, C.R.: Constructive and composite heuristic solutions to the P//$\sum C_i$ scheduling problem. European Journal of Operational Research 132, 439–452 (2001)

Merkle, D., Middendorf, M.: An ant algorithm with a new pheromone evaluation rule for total tardiness problems. In: Oates, M.J., Lanzi, P.L., Li, Y., Cagnoni, S., Corne, D.W., Fogarty, T.C., Poli, R., Smith, G.D. (eds.) EvoIASP 2000, EvoWorkshops 2000, EvoFlight 2000, EvoSCONDI 2000, EvoSTIM 2000, EvoTEL 2000, and EvoROB/EvoRobot 2000. LNCS, vol. 1803, pp. 287–296. Springer, Heidelberg (2000)

Minella, G., Ruiz, R., Ciavotta, M.: A review and evaluation of multi-objective algorithms for the flowshop scheduling problem. INFORMS Journal on Computing published online before print, April 2 (2008)

Miyazaki, S., Nishiyama, N.: Analysis for minimizing weighted mean flowtime in flowshop scheduling. Journal of the Operations Research Society of Japan 23, 118–132 (1980)

Miyazaki, S., Nishiyama, N., Hashimoto, F.: An adjacent pairwise approach to the mean flowtime scheduling problem. Journal of Operations Research Society of Japan 21, 287–299 (1978)

Murata, T., Ishibuchi, H., Tanaka, H.: Multi-objective genetic algorithm and its applications to flowshop scheduling. Computers & Industrial Engineering 30, 957–968 (1996)

Nawaz, M., Enscore Jr, E.E., Ham, I.: A heuristic algorithm for the m-machine, n-job flowshop sequencing problem. OMEGA 11, 91–95 (1983)

Pasupathy, T., Rajendran, C., Suresh, R.K.: A multi-objective genetic algorithm for scheduling in flow shops to minimize the makespan and total flow time of jobs. International Journal of Advanced Manufacturing Technology 27, 804–815 (2006)

Rajendran, C.: Two-stage flowshop scheduling problem with bicriteria. Journal of the Operational Research Society 43, 871–884 (1992)

Rajendran, C.: Heuristic algorithm for scheduling in a flowshop to minimize total flowtime. International Journal of Production Economics 29, 65–73 (1993)

Rajendran, C.: A heuristic for scheduling in flowshop and flowline-based manufacturing cell with multi-criteria. International Journal of Production Research 32, 2541–2558 (1994)

Rajendran, C.: Heuristics for scheduling in flowshop with multiple objectives. European Journal of Operational Research 82, 540–555 (1995)

Rajendran, C., Ziegler, H.: Ant-colony algorithms for permutation flowshop scheduling to minimize makespan / total flowtime of jobs. European Journal of Operational Research 155, 426–438 (2004)

Rajendran, C., Ziegler, H.: Two ant-colony algorithms for minimizing total flowtime in permutation flowshops. Computers & Industrial Engineering 48, 789–797 (2005)

Ruiz, R., Stuetzle, T.: A simple and iterated greedy algorithm for the permutation flowshop scheduling problem. European Journal of Operational Research 177, 2033–2049 (2007)

Ruiz, R., Maroto, C., Alcaraz, J.: Two new robust genetic algorithms for the flowshop scheduling problem. OMEGA 34, 461–476 (2006)

Sridhar, J., Rajendran, C.: Scheduling in flowshop and cellular manufacturing systems with multiple objectives – a genetic algorithmic approach. Production Planning & Control 7, 374–382 (1996)

Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2, 221–248 (1994)

Stuetzle, T.: An ant approach to the flow shop problem. In: Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT 1998), Verlag Mainz, Aachen, pp. 1560–1564 (1998)

Taillard, E.: Benchmarks for basic scheduling problems. European Journal of Operational Research 64, 278–285 (1993)

Tasgetiren, M.F., Liang, Y.-C., Sevkli, M., Gencyilmaz, G.: A particle-swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. European Journal of Operational Research 177, 1930–1947 (2007)

T'kindt, V., Billaut, J.-C.: Multicriteria scheduling: Theory, models and algorithms. Springer, Berlin (2002)

T'kindt, V., Monmarche, N., Tercinet, F., Lauegt, D.: An ant colony optimization algorithm to solve a two-machine bicriteria flowshop scheduling problem. European Journal of Operational Research 142, 250–257 (2002)

Varadharajan, T.K., Rajendran, C.: A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs. European Journal of Operational Research 167, 772–795 (2005)

Widmer, M., Hertz, A.: A new heuristic method for the flowshop sequencing problem. European Journal of Operational Research 41, 186–193 (1989)

Wang, C., Chu, C., Proth, J.-M.: Heuristic approaches for n/m/F/$\sum C_i$ scheduling roblems. European Journal of Operational Research 96, 636–644 (1997)

Woo, H.S., Yim, D.S.: A heuristic algorithm for mean flowtime objective in flowshop scheduling. Computers and Operations Research 25, 175–182 (1998)

Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. IEEE Transactions on Evolutionary Computation 3, 257–271 (1999)