Int J Syst Assur Eng Manag (November 2017) 8(Suppl. 2):S1630–S1642
DOI 10.1007/s13198-017-0635-3

ORIGINAL ARTICLE

Integrated production scheduling and maintenance planning in a hybrid flow shop system: a multi-objective approach

Mostafa Zandieh[1] · **Seyed Mojtaba Sajadi**[2] · **Reza Behnoud**[3]

Received: 24 May 2016 / Revised: 14 April 2017 / Published online: 26 May 2017

Abstract This study deals with a hybrid flowshop system with sequence-dependent setup times. Two objectives have been considered. Minimizing makespan for production purpose along with minimizing unavailability of the system for maintenance purpose are the objectives of this problem. Two meta-heuristics have been developed for the research problem. First one is a non-dominated sorting genetic algorithm-II (NSGA-II), while the second one is a hybridized NSGA-II (HNSGA-II), which is accompanied by a local search procedure to create better results. These two algorithms allow the decision maker to find compromise solutions between production objectives and preventive maintenance ones. Two decisions should be taken at the same time: finding the best assignment and sequence of jobs on machines in order to minimize the makespan, and deciding how often to perform preventive maintenance actions in order to minimize the system unavailability. Three approaches have been suggested for evaluation and comparison the efficiency of algorithms. The results indicate that the HNSGA-II presents better solutions compared to the ordinal NSGA-II in terms of objective functions viewpoint, while the results are obviously reversed in balance degree of achieving both objectives simultaneously.

Keywords Hybrid flowshop scheduling · Preventive maintenance · Sequence-dependent setup times · Unavailability of production system

1 Introduction

Machines are the main sources of production scheduling in manufacturing industries. Two services are directly in contact with machines, i.e., production and preventive maintenance. Scheduling comprises of allocating determinate sources to a set of jobs to optimize a definite objective function such as minimizing the tardiness/earliness of jobs, minimizing the job completion times, and minimizing the job processing times. Regarding to scheduling, many studies have been done with respect to different shop environments (i.e., single machine, parallel machines, flowshop, job shop, open shop, and hybrid systems), objective functions, and also the problem requirements and restrictions (i.e., setup times, postponed orders, etc.) (Berrichi et al. 2009). Most of Scheduling problems are NP-Hard (Garey and Johnson 1979).

In the literature, it has been mostly assumed that machines are accessible. However, in actual production systems, machines may be unavailable at certain times due to preventive maintenance activities, failure, etc. The most important activities of repair scheduling service and especially preventive maintenance is to prepare an appropriate preventive maintenance program to optimize a definite objective function such as maintenance costs or keeping machines at a stable situation permanently. At last decades,

✉ Mostafa Zandieh
m_zandieh@sbu.ac.ir

[1] Department of Industrial Management, Management and Accounting Faculty, G.C., Shahid Beheshti University, Tehran, Iran

[2] Faculty of Entrepreneurship, University of Tehran, Tehran, Iran

[3] School of Industrial Engineering, Faculty of Engineering, Science and Research Branch of Islamic Azad University, Tehran, Iran

although widespread researches have been done to solve such problems, most of them do not consider producing necessities (Berrichi et al. 2009).

Despite the dependency between scheduling and preventive maintenance programming, these activities are programmed and operated separately in real production systems. For years, the relation between production and preventive maintenance has been considered as a complex management decision problems. This complexity, if predetermined maintenance periods are not observed in production process, may result in interruptions like unexpected repairing or machine failures. This being the case, production disorder and unsatisfied demand in production might be followed. Thus, it is an important necessity to have a novel pattern and method for a model in which scheduling and preventive maintenance programming are considered simultaneously, in order to make a conjunction between them. Therefore, predeterminant solutions lead to the best results of production and maintenance activities by together. This is the main motivation of the research problem covered the gap between this research and previous studies.

Specifications of hybrid flowshop used in this research are defined as below:

- n independent jobs must be scheduled as a flow through s stages. Each job must be operated through stage $1, 2, \ldots, s$ (no job skipping).
- Stage g has $m^g (g = 1, 2, \ldots, s)$ parallel machines $(m^g \geq 1)$. At every stage, jobs can be processed by at least one machine, but the processing time of a job can vary in terms of different machines. Machines of a bottleneck stage are identical and/or unrelated.
- The processing time of job $i (i = 1, 2, \ldots, n)$ on machine $k (k = 1, 2, \ldots, m)$ at stage g is represented by p_{ik}^g. Job processing time is constant and independent to the job sequence. When machines are identical, processing time is depended on the job and the stage, i.e., p_i^g.
- All jobs and machines are available at the beginning of the planning horizon.
- Job splitting is not allowed. In other words, when a job is being operated on a machine, the job cannot be taken from the machine till it is finished.
- Jobs are permitted to wait between two stages. Also, the capacity of storage between two stages is assumed to be unlimited.
- The failure rate of machine k at stage g is equal to λ_k^g and repair rate of machine k at stage g is assumed as μ_k^g. Hence the preventive maintenance time and unavailability of the system are computable.

The paper is organized as follows: Sect. 2 presents a literature review of scheduling problems taking into account sequence-dependent setup times and preventive maintenance. Section 3 presents the problem description, proposed integrated model of joint production and preventive maintenance in hybrid flowshop, and how to calculate system unavailability. In Sect. 4, the proposed algorithms used to solve the problem have been introduced. Section 5 presents the computational results which contains data generation, parameter tuning, comparison metrics, experimental results, and the results analysis. Section 6 states conclusions and some future research opportunities in the research problem.

2 Literature review

In order to simplify the scheduling problems, the setup times are seldom considered, however in real industries, the processing and setup times are mostly independent. Gupta and Tunc (1991) have suggested four heuristic algorithms to minimize makespan (C_{max}) for hybrid flowshops, assuming that the setup and transportation times are apart. Another characteristic of this research is that the setup times a sequence-dependent. Considering the defects resulted by this assumption, researches over this context were rare. Over this, we can nominate Kurz and Askin's (2004) study. In this paper, an integer programming and random keys genetic algorithm (RKGA) system are used.

Hadidi et al. (2012a) considered the integration of production scheduling and preventive maintenance scheduling on a single machine including random failures. The objective function was to minimize the total weighed expected job completion times. They developed a mixed-integer programing model to jointly consider production scheduling and maintenance decisions. Berrichi et al. (2009) discussed two meta-heuristics to optimize joint production and preventive maintenance scheduling problem in parallel machines system. They aimed to minimize the makespan for the production part and to minimize the system unavailability for the maintenance part. To solve this problem, they developed two meta-heuristics, weighted sum genetic algorithm (WSGA) and non-dominated sorting genetic algorithm-II (NSGA-II). Lastly they compared the results of these algorithms and found out that NSGA-II has better performance. Yulan et al. (2008) expanded a multi-objective integrated optimization research to solve preventive maintenance planning and production scheduling problems for a single machine environment. They considered five objectives simultaneously. Multi-objective genetic algorithm (MOGA) was used to solve this model. For different aspects of

integration in production planning and scheduling, maintenance and quality, readers are referred to Hadidi et al. (2012b).

Naderi et al. (2009a) have investigated a single objective hybrid flowshop problem with sequence-dependent setup times and multiple preventive maintenance strategies. The optimization criterion was to minimize the makespan. Their work divides into two parts: (1) because the integrated methods proposed in the literature are often complicated and problem-specific; they offered a simply implementable technique. (2) To solve the problem, they have proposed a novel variable neighborhood search (VNS) and the adaptations of some accessible high performing meta-heuristics in the literature. The proposed VNS uses advanced neighborhood search structures. A sample problem has been established to carefully evaluate the algorithms. All the results illustrated that the VNS presents better performance in comparison with the other algorithms. In another study, Naderi et al. (2009b) have proposed two techniques for a single objective job shop scheduling with sequence-dependent setup times and preventive maintenance policies which is simplistically adaptable to any other machine scheduling problem. The optimization criterion was to minimize the makespan. To solve the problem, four meta-heuristics has been implemented. Performance evaluation of proposed algorithms has been done by comparing their results through two sets of benchmarks.

Kaabi et al. (2002) have studied the single machine and permutation flowshop situations in two researches. In these researches, the maintenance periods must be performed in pre-identified intervals and production scheduling is jointed to maintenance planning. Allaoui et al. (2008) have considered joint production scheduling and preventive maintenance planning for two machine flowshops with n jobs, with the objective of minimizing makespan. They assume that the maintenance activities must be performed on jobs at the first T periods of scheduling and the jobs cannot be split. After offering some characterizations of optimum solutions of this problem, they indicated that this is a NP-hard problem. At last, they focused on optimized solutions in some situations. As seen, the system was not hybrid flowshop here, it is two machine flowshop; furthermore, despite it is a NP-hard problem, they seek the solution by simplifying assumptions and considering the small-size problem.

Marett and Wright (1996) compared Simulated Annealing and Tabu-Search and utilized the comparison to solve large and complicated multi-objective problems in flowshop environment. Naderi et al. (2011) studied incorporating periodic preventive maintenance into a flexible flowshop scheduling problem with the aim of minimizing the makespan. They proposed two meta-heuristics including genetic algorithm and artificial immune system to deal with the research problem along with some constructive heuristics to tackle the problem.

Loukil et al. (2005) developed a multi-objective simulating algorithm for multi-objective production scheduling problem in three different environments of single machine, parallel machines, and flowshop considering seven objectives. Zandieh and Karimi (2011) developed a multi-population genetic algorithm (MPGA) for multi-objective group scheduling problems in a hybrid flow shop environment where setup times are dependent on sequence. The objective function was to minimize the total weighted tardiness and the maximum completion time of jobs simultaneously.

Angelo-Bello et al. (2011) considered the problem of scheduling jobs on a single machine with programmed periodic preventive maintenance actions and sequence-dependent setup times. They claimed that such problem has been considered in operations research literature. Therefore, they presented a solution based on metaheuristic procedures to convey high quality solution in appropriate computational times. Kapalanoglu (2014) considered job scheduling on a single machine with sequence-dependent setup times and maintenance activities in dynamic manufacturing environment (which order or job release time is dynamic to the system). The problem has been solved using multi-agent systems under the condition of regular and irregular maintenance on the machine. The experiments results showed that the method presents acceptable solutions. Hadidi et al. (2015) studied the practical implications of managerial decisions to integrate the scheduling of production and maintenance operations with the purpose of minimizing product holding costs and maintenance costs. In addition, Hadidi and Rahim (2015) studied a sequential imperfect preventive maintenance model and extended it to consider multiple units. The purpose of the model was to optimize the maintenance management process for multiple-unit systems.

3 Problem description

We consider hybrid flowshop with sequence-dependent setup times and unrelated-parallel machines in some stages for production aspect. The makespan should be minimized. We assume that all jobs are available at the beginning of the planning horizon. Two activities are performed in every stage of hybrid flowshop. First, the job assignment to machines; second, the job sequence on each machine. Numerous studies have been devoted in this field.

We focus on systematic preventive maintenance (PM) to consider preventive maintenance necessities. PM activities keep machines in the good working condition (i.e., it

increases the system availability), and decrease the system costs by preventing unexpected failures. The problem is to determine PM dates to decrease the unavailability of each machine. Availability of a system is defined as "the probability that a system or a component is performing a required function at a given point in time or over a stated period of time when operated and maintained in a prescribed manner" (Ebeling 1997). So, the availability of machine g in stage k, M_k^g, is the probability of the proper operating of machine M_k^g at time $t\big(A_k^g(t) = P\big(M_k^g$ *is working at time* $t\big)\big)$. The opposite of availability is known as unavailability $(\bar{A}_k^g(t) = 1 - A_k^g(t))$.

The availability of a machine M_k^g is defined by its repair rate μ_k^g and failure rate λ_k^g, which are considered to be constant in this paper. We assume that two exponential probability distributions with parameters λ_k^g and μ_k^g, respectively, define the failure and repair of a machine M_k^g. We also assume that PM activities bring machines back to "as good as new" condition. Considering these assumptions, system availability at time $t = 0$ is defined as follows (Ebeling 1997; Villemeur 1991):

$$A_k^g(t) = \frac{\mu_k^g}{\lambda_k^g + \mu_k^g} + \frac{\lambda_k^g}{\lambda_k^g + \mu_k^g} exp\big[-\big(\lambda_k^g + \mu_k^g\big)t\big].$$

Increasing the time causes reduction in system availability and, consequently, the unavailability will increase. If we consider PT as the completion time of a PM activity on machine M_k^g, system availability at time t is defined as follows (Ebeling 1997; Villemeur 1991):

$$A_k^g(t) = \frac{\mu_k^g}{\lambda_k^g + \mu_k^g} + \frac{\lambda_k^g}{\lambda_k^g + \mu_k^g} E(t),$$
$$\text{where } E(t) = exp\big[-\big(\lambda_k^g + \mu_k^g\big)(t - PT)\big].$$

Availability of a system depends on the system type and its components characteristics. Here, unavailability of each stage g, $\bar{A}^g(t)$, with m^g parallel machines at time t is defined by the following expression:

$$\bar{A}^g(t) = \prod_{k=1}^{k=m^g} \big[1 - A_k^g(t)\big].$$

Consequently, we use the following equation to calculate the unavailability of entire hybrid flowshop system with k stages:

$$\bar{A}_{system} = 1 - \prod_{g=1}^{g=s} \Big[1 - \max_t \{\bar{A}^g(t)\}\Big].$$

The integrated model considers two objectives simultaneously: Minimization of makespan for production aspect, and minimization of system unavailability for preventive maintenance aspect. Therefore, two different decisions should be taken at the same time. First, finding best assignment and sequence of n jobs on m^g machines in stage g through s successive stages, in order to minimize makespan; second, deciding when is the best time to perform PM activities, in order to minimize system unavailability. Although both objectives deal with total efficiency of the system, they are in conflict so that optimizing one of them will make the other one worse. In other words, performing PM actions will decrease system unavailability while they increase the makespan at the same time. Conversely, not performing PM actions will deliver reverse results. Let C_i^g be the makespan of job i in stage g, and C_{max}^g is the completion time of the last operated job in stage g (makespan): $C_{max}^g = max_{i=1,\ldots,n}^g\{C_i^g\},\ g = 1, \ldots, s.$

Let $T = \big\{0, st_1^g, st_2^g, \ldots, st_m^g, C_{max}^g\big\}, g = 1, \ldots, s$ where $st_1^g, st_2^g, \ldots, st_m^g$ are starting times of PM activities on all machines in stage g. Since unavailability is an increasing function in $\big[st_k^g, st_{k+1}^g\big]$, $k = 1, \ldots, m$, assuming that $st_0^g = 0$ and $st_{m+1}^g = C_{max}^g$, and according to this assumption that after performing PM activities the machine will be in "as good as new" condition, then the unavailability of the system will be only calculated at $st_1^g, st_2^g, \ldots, st_{m+1}^g$. Processing time of a PM activity on machine M_k^g is the average of preventive maintenance activity, which equals to $1/\mu_k^g$ (Adzapka et al. 2004). The two objectives of the problem that should be considered under the defined constraints are:

$F_1 = C_{max}$, the makespan.

$F_2 = \big\{\bar{A}_{system}\big\}$, the unavailability of the system.

4 Proposed algorithms

A meta-heuristic procedure is required for solving the research problem, particularly for industry-size problems. The meta-heuristic algorithm might be based on a local search structure such as tabu search (Shahvari et al. 2009, 2012; Shahvari and Logendran 2015, 2017), a population-based structure such as genetic algorithm (Zandieh et al. 2010) and particle swarm optimization (Hajinejad et al. 2011), and a hybrid of both structures such as tabu search/path-relinking (Shahvari and Logendran 2016a, b). Since two objectives are minimized simultaneously in the research problem, there is a set of solutions instead of a single optimal solution. In addition, the feasible solution area of the proposed research problem is larger than regular hybrid flow shop scheduling problems. Based on aforementioned reasons, our preliminary results indicate the better performance of meta-heuristic algorithms accompanied by population-based structures compared to the ones accompanied by local search structures. The reason lies in the fact that a population-based structure providing a set of

solutions at each iteration is capable to present good quality solutions in less computational time, compared to a local search structure. Two genetic algorithms have been developed and their results have been compared. The first algorithm is the known non-dominated sorting genetic algorithm-II (NSGA-II) which is based on domination concept. The second one is 2-OPT NSGA-II that have been developed for the first time in this study, and is a kind of hybridized NSGA-II. It tries to improve the first algorithm by considering a local search structure in NSGA-II.

In multi-objective optimization with two minimization objective functions (f_1, f_2), for both decision vectors x and y, we express that x dominates y $(x \prec y)$ if $f_1(x) < f_1(y)$ and $f_2(x) \leq f_2(y)$ or $f_1(x) \leq f_1(y)$ and $f_2(x) < f_2(y)$. Derived non-dominated solution set by an evolutionary algorithm is called Pareto front.

4.1 NSGA-II

NSGA-II is defined as an elitist multi-objective evolutionary algorithm which calculates an approximation of non-dominated solution set in terms of the domination concept. To identify the non-dominated fronts, a ranking procedure has been applied in every generation of the algorithm. NSGA-II has been organized in comparison with most comparable optimization algorithms by numerous studies in the research literature. (Basseur 2006; Coellom and Cortes 2002; Deb et al. 2000; Gaspar-Cunta and Covas 2003; Ishibuchi et al. 2003). The following fields are some applications of this algorithm: polymer extrusion optimization in chemistry (Gaspar-Cunta and Covas 2003), vehicle routing optimization (Velasco et al. 2006), scheduling optimization (Landa-Silva et al. 2003; Vilcot et al. 2006), and supply chain optimization (Amodeo et al. 2007). The general procedure of NSGA-II is presented by a pseudo-code in Fig. 1.

First, a random initial population $(SeqSet_0)$ that includes N random sequences of job 1 to n is created of size N. This population is sorted in several fronts based on the domination concept. In other words, the objective functions of each produced sequence is calculated after passing through hybrid flowshop based on SPTCH (Sect. 4.1.6) and is sorted in the related Pareto front. Each solution is evaluated by its non-dominated level (first level is the best). Then two point crossover and binary tournament selection operators is used to create offspring set $(OffSeqSet_0)$ of size N. For generations $l \geq 1$, the procedure is different. The first phase is to create a population $TotSeqSet_l = SeqSet_l \cup OffSeqSet_l$ of size $2N$ and implement non-dominated sorting procedure (ranking) for producing a list of non-dominated fronts. In the second phase, a new population of sequences $SeqSet_{l+1}$ including N best solutions of $TotSeqSet_l$ is created until the number of solutions in $SeqSet_{l+1}$ is less than N. To complete $SeqSet_{l+1}$ with $N - |SeqSet_{l+1}|$ solutions, the crowding distance procedure is applied to the first front. Then, the population $SeqSet_{l+1}$ is used to create offspring set of new sequence $OffSeqSet_{l+1}$ of size N by the selection and crossover operators.

Algorithm (1): NSGA-II
1: Create initial populations $SeqSet_0$ and $OffSeqSet_0$ of size N
2: **While** stopping criteria are not verified **do**
3: Create population $TotSeqSet_l = SeqSet_l \cup OffSeqSet_l$
4: Construct the different fronts $Front_j$ of $TotSeqSet_l$ by the *non-dominated sorting procedure*
5: Put $SeqSet_{l+1} = \emptyset$ and $j = 0$,
6: **While** $|SeqSet_{l+1}| + |Front_j| < N$ **do**
7: $SeqSet_{l+1} = SeqSet_{l+1} \cup Front_i$
8: $j = j + 1$
9: **End While**
10: Include in $SeqSet_{l+1}$ the $(N - |SeqSet_{l+1}|)$ of $Front_i$ according to the *crowding distance procedure*
11: Create $OffSeqSet_{l+1}$ from $SeqSet_{l+1}$ by selection and crossover
12: **End While**

Fig. 1 NSGA-II pseudo code

4.1.1 Non-dominated sorting

Applied on a population $SeqSet$, non-dominated sorting procedure presents a list of non-dominated fronts. Two parts are calculated for each solution: nd_u which is the number of solutions that dominate solution u and D_u which is a set of solutions that solution u dominates them. If $nd_u = 0$, then solution u belongs to the first Pareto front, so the solutions of this front will be eliminated from the solutions set, and the same procedure will be applied on the remaining solutions until all solutions will be categorized in their related front (Deb et al. 2000).

4.1.2 Crowding distance

The average distance of two points close to a specific point is calculated based on each objective viewpoint in order to estimate the density of solutions close to a solution (sequence). This procedure is known as "crowding distance". If a solution belongs to a better (less) front, it will be selected according to non-dominated sorting procedure, while if two solutions belong to the same front, the solution with a larger crowding distance is preferable for the next generation. In other words, the solutions in less crowded area are preferable (Deb et al. 2000).

4.1.3 Representation scheme

Every solution or chromosome contains two parts: production part and preventive maintenance part. The number of genes in production part equals to the number of jobs. A random number is generated in every gene containing two parts itself. The first part is a natural number that represents the number of processing machines so that the job in this gene will be assigned to the machine with

this number. The second part appeared after the natural number is a decimal amount that determines the sequence of assigned jobs to each machine. If the decimal part of a gene is smaller than another one, it has priority in processing on the machine. Therefore, the generated random numbers will be placed in interval $[1, m+1)$. The number of genes in the preventive maintenance part equals to the number of machines. Each gene contains a natural number that is representative of preventive maintenance activities periods. Periods of PM is generated randomly from interval $\left[p^g_{Short,k}, C^g_{Late,k}\right]$. $p^g_{Short,k}$ is the processing time of the earliest job assigned to machine k in stage g, while $C^g_{Late,k}$ is the completion time of the latest job assigned to machine k in stage g (Berrichi et al. 2009). For example, (16, 20) with two machines indicates that PM activity should be performed every 16 time units on the first machine, while it should be performed every 20 time units on the second machine. These chromosomes will be generated in each stage and main chromosome of a problem in this study contains as much explained chromosomes as the number of stages.

As it can be seen in Fig. 2, we assume that we have a hybrid flowshop system with two machines in a stage and there are totally 6 jobs in the system. In the production part of the chromosome, there is 6 genes that each one represents the job of its number. As an example, the first gene is the representative of the first job of the system. Hence regarding Fig. 6 and given definitions, jobs 1–6 are assigned to machines 1, 2, 1, 2, 2, and 1, respectively. Considering the decimal part of numbers in the genes, the sequence of jobs is determined on each machine. In other words, jobs on machine 1 is processed in sequence 1, 3, and 6 ($0.23 < 0.45 < 0.96$), while jobs on machine 2 is processed in sequence 2, 5, and 4 correspondingly ($0.03 < 0.37 < 0.89$).

Preventive maintenance activities are performed based on the rational strategy. Let St^g_{ik} and C^g_{ik} are start time and completion time of job I, respectively, on machine k in stage g. Let PT_e be the expected time to satisfy the preventive maintenance activity. If $C^g_{ik} - PT_e \geq PT_e - St^g_{ik}$, then PM activity is performed on St^g_{ik}; otherwise it is performed on C^g_{ik}.

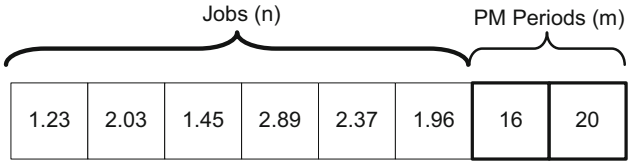

Fig. 2 Chromosome of one stage with 6 jobs and 2 machines

4.1.4 Two-point crossover

Initial population is generated randomly. Several crossover operators related to genetic algorithm have been examined and finally two-point crossover has been chosen since it presents good quality results (Ishibuchi et al. 2003). Stopping criteria is a fixed number of iterations.

For both production and preventive maintenance parts, two-point crossover operators have been applied. The first offspring is generated by keeping the first and last part of the chromosome related to the first parent along with substituting emitted jobs based on their priority in the second parent. The same method is applied to generate the second offspring. Two-point crossover method have been depicted in Fig. 3.

4.1.5 Binary tournament selection

According to former descriptions, an initial population is generated randomly. In other words, the numbers of each gene are generated randomly in the first generation. Though the procedure of generating and placing these numbers in the next generation is different and performed by binary tournament selection and crossover operators.

Although binary tournament selection is like the tournament selection, the size of tournament solutions group is two. Thus, two solutions are selected randomly from the population and the objective functions of both solutions are calculated. If one of the solutions dominates the other one, it will be transferred to the next generation; otherwise if the solutions are non-dominated ones, they will be ignored and another tournament selection will take place.

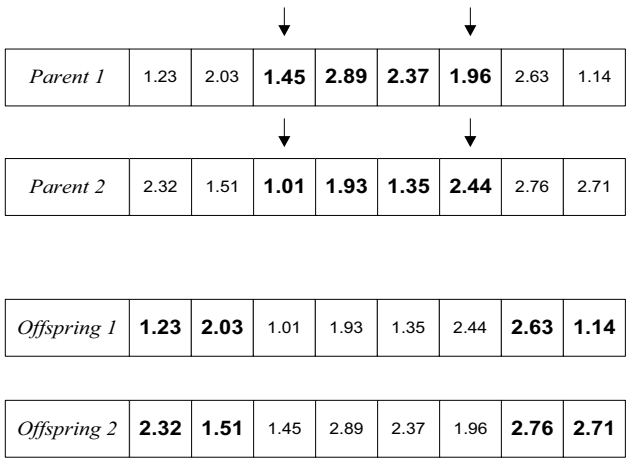

Fig. 3 Two-point crossover

4.1.6 SPT cyclic heuristic

Since the assignment and sequence of jobs on machines might be different in each stage, the SPT cyclic heuristic (SPTCH) is utilized as Fig. 4. In this method, jobs are sorted at the first stage on the bases of ascending order of modified processing times, $\tilde{p}_i^g$. The modified processing time of job i in stage g is defined as: $\tilde{p}_i^g = p_i^g + min_y s_{yi}^g$, where s_{yi}^g is required setup time to process job i after job y in stage g, and p_i^g is the processing time of job i in stage g (Shahvari and Logendran 2017, 2016b; Kurz and Askin 2004). We use this heuristic in order to sort the jobs based on their earliest ready times in the following stages.

4.2 HNSGA-II

In this study, a competitive algorithm for NSGA-II have been developed to obtain a better performance than a ordinal NSGA-II. The mechanism of this competitive algorithm can assure an improvement in the performance of a ordinal NSGA-II. Some studies have been perused on hybridized NSGA-II. For instance, in one of these studies, Pareto hill climbing NSGA-II (PHC-NSGA-II) have been developed (Bechikh et al. 2008). They applied a hill climbing local search on the front solutions with respect to the crowding distance, and then improved solutions is added to the front. They implemented a novel crowding distance (in terms of the decision space, not the objective space) to generate the next generation. They proved that their algorithm functions better than the ordinal NSGA-II. In another study, a neighboring-max crossover is accompanied by the ordinal NSGA-II and, consequently, the population distribution of Pareto front is improved. This algorithm is known as hybridization encouraged mechanism based NSGA-II (HEM-Based NSGA-II) (Yijie and Gongzhang 2008).

Algorithm (2): SPT Cyclic Heuristic
1: Generate modified processing times $\tilde{p}_i^1$.
2: Order the jobs in non-decreasing order (SPT) of $\tilde{p}_i^1$.
3: At each stage $g = 1, \ldots, s$, assign job 0 to each machine in that stage.
4: **For** stage 1 **do**
  a. **Let** $bestmc = 1$.
  b. **For** $[i] = 1$ to n, $i \in S^1$ **do**
    For $mc = 1$ to m^1 **do**
      Place job $[i]$ last on machine mc.
      Find the completion time of job $[i]$. **If** this time is less on mc than $bestmc$ **then**
        Let $bestmc = mc$.
    Assign job $[i]$ to the last position on machine $bestmc$.
5: **For** each stage $g = 2, \ldots, s$ **do**
  a. Update the ready times in stage g to be the completion times in stage g - 1.
  b. Arrange jobs in increasing order of ready times.
  c. **Let** $bestmc = 1$.
  d. **For** $[i] = 1$ to n, $i \in S^g$ **do**
    For $mc = 1$ to m^g **do**
      Place job $[i]$ last on machine mc.
      Find the completion time of job $[i]$.
      If this time is less on mc than on $bestmc$ **then**
        Let $bestmc = mc$.
    Assign job $[i]$ to the last position on machine $bestmc$.

Fig. 4 SPTCH pseudo code

4.2.1 2-OPT local search

We developed a novel method for hybridized NSGA-II in this study. In the ordinal NSGA-II, we only use the crowding distance to select the last solutions of the next generation, while in our presented algorithm, first a local search applies on all chromosomes and then the obtained solutions are added to the solution set. Then, the next generation is determined using non-dominated sorting and crowding distance.

We have inspired our suggested local search heuristic from 2-OPT local search method. In the 2-OPT method, two points are selected and all genes between these two points is inversed (Nilsson 2003). In spite of inversing all genes between two selected points, in our suggested 2-OPT, only two selected points will be replaced. Thus, the assignment of jobs are constant and only the job sequence will be changed by this method. Hence only the decimal part of numbers in each gene is replaced and the natural part of numbers is stable. This mechanism is applied on all pair of genes and on all chromosomes and, consequently, the new solutions are added to the solution set. Afterwards using the non-dominated sorting and the crowding distance sorting population, the next generation will be generated. Figure 5 depicts the method of using local search heuristic.

As it is shown in Fig. 6, the new HNSGA-II differs with the ordinal NSGA-II only in line 4 of pseudo-code. In other words, the local search is applied on all solutions

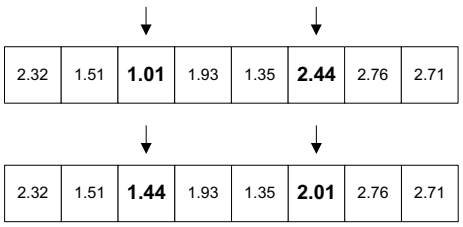

Fig. 5 Modified 2-OPT local search

Algorithm (3): HNSGA-II
1: Generate initial populations $SeqSet_0$ and $OffSeqSet_0$ of size N
2: **While** stopping criteria are not verified **do**
3: Generate population $TotSeqSet_l = SeqSet_l \cup OffSeqSet_l$
4: Implement Modified 2-OPT Local Search on every solution and add the results to $TotSeqSet_l$
5: Construct the different fronts $Front_j$ of $TotSeqSet_l$ by the *non-dominated sorting procedure*
6: Put $SeqSet_{l+1} = \emptyset$ and j = 0,
7: **While** $|SeqSet_{l+1}| + |Front_j| < N$ **do**
8: $SeqSet_{l+1} = SeqSet_{l+1} \cup Front_j$
9: j = j + 1
10: **End While**
11: Include in $SeqSet_{l+1}$ the $(N - |SeqSet_{l+1}|)$ of $Front_j$ according to the *crowding distance procedure*
12: Generate $OffSeqSet_{l+1}$ from $SeqSet_{l+1}$ by selection and crossover
13: **End While**

Fig. 6 Pseudo code of HNSGA-II

(sequences) and then the next phases of the algorithm are followed.

5 Computational results

5.1 Data generation

In this research, Table 1 has been used to generate test problems in order to evaluate the performance of developed algorithms.

The number of jobs, processing times of jobs, the number of workstations (stages), the number of machines in each stage, the failure rate of machines, and the repair rate of machines have been shown in Table 1. Setup times are calculated as 20–40% of processing times so that the range of them will be [12, 24] (Rios-Mercado and Bard 1998). To validate bi-objective problems, 20 problems have been generated with the help of the aforementioned setting for parameters.

5.2 Parameter tuning

The performance of meta-heuristics has a straight relevance with setting parameter of them, so that a wrong parameter selection for an efficient algorithm might cause a total failure of algorithm. For this purpose, various methods have been suggested in the literature in which most of them are tentative. Parameter setting in the multi-objective algorithms is very difficult particularly when there are numerous parameters for the algorithm (Berrichi et al. 2009). We used a specialized experimental method for setting parameters of the algorithms. In our method, an appropriate combination of parameters for each kind of problem is used. Three levels of job numbers (6, 30 and 100) along with three levels of stage numbers (2, 4 and 8) and three different levels of machine distribution in each stage have been considered. Both developed algorithms have been run 5 times for each of the above combinations. In other words, each algorithm has been run 135 times. For all problems with any size, the crossover percentage is $Percent_c = 0.8$ since it has been experimentally proved that this amount of percentage leads to the best performance of both algorithms for all problems. In both algorithms, the population size is 200 for problems including 6 jobs. For problems including 30 jobs, the population size is 100 for 2 and 4 stages, while it is 50 for 8 stages. Finally, the population size of 20 have been considered for problems including 100 jobs due to prevent unacceptable increase of run time. Most usual stopping criteria of genetic algorithms is based on a fixed number of iterations. We also considered the specialized fixed number of iterations as a stopping criterion according to the problem size and run time. For 6 job problems, the fixed number of 100 iterations, for 30 job problems with 2 and 4 stages, the fixed number of 100 iterations, for 30 job problems with 8 stages, the fixed number of 50 iterations, and for 100 job problems, the fixed number of 35 iterations have been considered as stopping criteria.

5.3 Comparison metrics

To evaluate the operational performance of multi-objective algorithms, we use three methods and time elements. To explain these methods, we consider PF_{Known} as the final non-dominated solutions set of algorithm and PF_{True} as the optimum solutions set (or the best known solution) of the algorithm. We might not have PF_{True} for some multi-objective problems since deriving PF_{True} is impossible.

5.3.1 Mean ideal distance (MID)

Van Veldhuizen and Lamont (1998) developed a method known as Generational Distance which calculates the distance between PF_{Known} and PF_{True}. In order to calculate this metric, we should have PF_{True}. However, sometimes in the multi-objective optimization, we might not have an access to this set. Therefore, the main difficulty of such a method is its dependency to the optimum solutions set. In this study, we use another method that it is not dependent on the optimum solutions of the problem. Hence it can be more general and applicable.

Table 1 Component levels for data generation

Components	Levels		
Number of jobs	6	30	100
Distribution of machines in a stage	Uniform distribution in [2, 5]	Uniform distribution in [2, 10]	
Number of stages	2	4	8
Processing times	Uniform distribution in [50, 70]	Uniform distribution in [20, 100]	
Setup times	Uniform distribution in [12, 24]		
Failure rate of machines	0.1	0.3	
Repair rate of machines	0.1	0.3	

$$MID = \frac{\sum_{u=1}^{U} DS_u}{H},$$

where U is the number of vectors in PF_{Known} and DS_u is the Euclidian distance between each member of set with the zero point derived from equation $\sqrt{f_{1u}^2 + f_{2u}^2}$. f_{hu} is hth objective function in uth Pareto solution vector. Smaller value of this metric is preferable (Behnamian et al. 2010).

5.3.2 Rate of achievement to both objectives simultaneously (RAS)

This method is based on distance too. If a solution vector only satisfies one objective, it is not a suitable solution from this metric viewpoint. Solutions generating a balance between both objectives are preferable. This metric is defined as below:

$$RAS = \frac{\sum_{u=1}^{U} \left(\left(\frac{f_{1u} - FI_u}{FI_u} \right) + \left(\frac{f_{2u} - FI_u}{FI_u} \right) \right)}{U},$$

where $FI_u = min\{f_{1u}, f_{2u}\}$. Smaller value of this metric is preferable (Behnamian et al. 2010).

5.3.3 Covered surface under both sets (CS)

This metric compares covered surface under both sets and delivers percentage of solutions that dominate other solutions. In this method, there is no need to have PF_{True}. It is defined as follows:

$$CS(X', X'') = \frac{\left| u'' \in Xu''; \forall \acute{u} \in \acute{X} : \acute{u} \geq u'' \right|}{|Xu|},$$

In this equation, two vectors of decision variable related to X' and X'' are shown. If $CS = 1$, it means that X'' dominates X' (Zitzler 1999).

5.4 Experimental results

In this study, both algorithms have been coded and run by C# compiler in Windows Vista Home Edition operating system, dual core 2.5 GHz, and 4 GB RAM. The results of running NSGA-II and HNSGA-II in hybrid flowshop with sequence-dependent setup times considering two objectives of minimizing makespan and unavailability of the system are compared.

5.4.1 Comparison among algorithms

Figure 7 shows the improvement progress of non-dominated solutions in Pareto front through different iterations of both developed algorithms by different colors and symbols. Red squares are representatives of the last iteration in relation with the best Pareto front and final best population of each algorithm. This progress has been run for 6 test problems and it is shown by graphical charts.

By an overview on the charts depicted in Fig. 7, it is obvious that non-dominated solutions of HNSGA-II are closer to the utopia point than non-dominated solutions of NSGA-II. Therefore, HNSGA-II presents better performance from the viewpoint of CS metric.

Table 2 shows obtained values of all comparison metrics for 270 solved test problems. As it is shown in Table 2, HNSGA-II obviously has a better performance compared to NSGA-II from the CS metric viewpoint and generates better solutions. In other words, the percentage of Pareto front solutions of HNSGA-II that dominate Pareto front solutions of NSGA-II is obviously more. It can be concluded that the solutions derived from the HNSGA-II algorithm are closer to utopia point than the solutions from ordinal NSGA-II algorithm.

In terms of MID metric viewpoint, HNSGA-II functions better than NSGA-II in 30 and 100 job problems (medium- and large-size problems), while in 6 job problems (small-size problems) NSGA-II functions better than HNSGA-II.

In terms of RAS metric viewpoint, it is obvious that NSGA-II functions better than HNSGA-II. In other words, solutions derived from HNSGA-II are more suited for only one objective despite both objectives simultaneously in comparison with NSGA-II.

Finally, from the Time metric viewpoint, the HNSGA-II functions better and need less computational time than NSGA-II in 6 job problems (small-size problems), while NSGA-II is faster than HNSGA-II in medium- and large-size problems (30 and 100 job problems). The processing time of the HNSGA-II algorithm increases 45% compared to NSGA-II in medium- and large-size problems.

5.5 Results analysis according to problem size

With the help of paired t test shown in Table 3, we study the development of rationalization of the two algorithms based on number of jobs (problem size) and the amounts of comparison metrics.

As it is shown in Table 3, the two developed algorithms have significant difference ($P_{value} = 0.033$) for medium-size problems (30 jobs) with confidence level of 95% from MID metric viewpoint, so that the development of two separate algorithms is rational. Also, in confidence level of 90%, the difference for large-size problems (100 jobs) is significant ($P_{value} = 0.091$).

From RAS metric viewpoint, the analysis is the same as MID metric. From CS metric viewpoint, the difference between two algorithms is significant for all problem sizes (small-, medium-, and large-size). From Time viewpoint, it

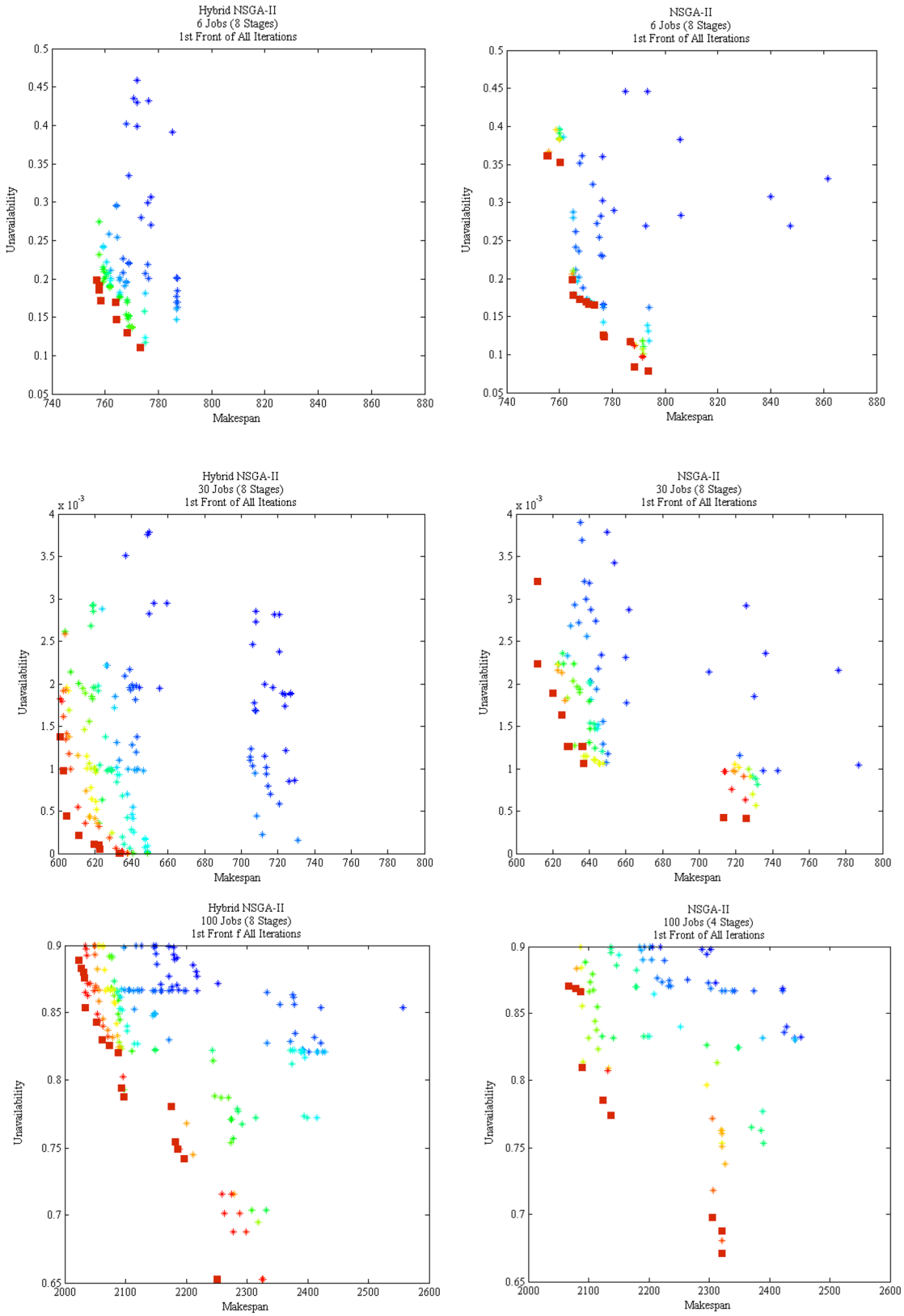

Fig. 7 Improvement progress of Pareto front through different iterations for 6 sample problems

Table 2 Evaluation of non-dominated solutions derived from performed algorithms considering size of problems and comparison metrics

Problem size		Algorithm and operational criterion							
Jobs	Stages	MID		RAS		CS × 1000		Time (s)	
		NSGA-II	Hybrid NSGA-II	NSGA-II	Hybrid NSGA-II	NSGA-II	Hybrid NSGA-II	NSGA-II	Hybrid NSGA-II
6 Jobs	2 Stages	0.8685	0.8902	6.1302	3.4686	0.0000	9.9568	21.58	21.13
	4 Stages	0.8355	0.832	11.2841	14.7161	0.0000	12.1957	74.23	86.31
	8 Stages	0.7581	0.8734	19.2274	21.4629	0.0000	20.356	566.79	505.82
Average of 6 jobs		0.8207	0.8652	12.2139	13.2159	0.0000	14.1695	220.87	204.42
30 Jobs	2 Stages	0.8583	0.6977	10.1355	44.9649	0.0000	2.4333	134.33	373.12
	4 Stages	0.814	0.7018	12.8839	33.5378	0.9667	6.6667	1006.88	2021.34
	8 Stages	0.888	0.8032	9.8965	26.6092	8.9467	13.7187	1463.01	3743.87
Average of 30 jobs		0.8534	0.7342	10.9720	35.0373	3.3045	7.6062	868.07	2046.11
100 Jobs	2 Stages	0.8743	0.6731	18.4329	21.5443	0.1179	4.3898	1472.56	2119.38
	4 Stages	0.9821	0.8063	20.0031	26.6728	4.4675	12.6667	6517.23	9896.61
	8 Stages	0.8635	0.8124	24.2837	35.4386	6.3476	15.6233	10,831.45	13,254.52
Average of 100 jobs		0.9066	0.7639	20.9066	27.8852	3.6443	10.8933	6273.75	8423.50
Total average		0.8603	0.7878	14.6975	25.3795	2.3163	10.8897	2454.23	3558.01

Table 3 *p* values of paired *t* test for both NSGA-II and HNSGA-II algorithms according to comparison metrics and problem size

Problem size	*p* values			
	MID	RAS	CS	Time
6 jobs	0.343	0.645	0.046	0.542
30 jobs	0.033	0.048	0.047	0.186
100 jobs	0.091	0.096	0.041	0.115

is obvious that there is not a significant difference between computational times of two algorithms and therefore HNSGA-II is rational due to the improvement in the algorithm performance in comparison with NSGA-II. Figure 8 depicts the aforementioned results.

6 Conclusions and future works

In this study two meta-heuristic algorithms have been developed for a bi-objective hybrid flowshop problem with sequence-dependent setup times. The objectives are to minimize makespan and unavailability of the system simultaneously. These objectives are considered jointly due to real industries' requirements. The problem is NP-Hard and therefore meta-heuristic algorithms can be utilized. The algorithms developed in this study have not been implemented to solve this kind of problems in the literature. Also, joint production and preventive maintenance scheduling for hybrid flowshop with sequence-dependent setup times is a new problem in the literature that have not been studied before. The developed algorithms are NSGA-II and HNSGA-II where in the second one, a 2-OPT local search procedure is accompanied by ordinal NSGA-II algorithm to improve the algorithm performance.

In NSGA-II, we focus on selecting better solutions with better objective function values which it is assured by selecting better Pareto fronts until reaching the population size limit. For selecting a part of solutions group in a Pareto front, the crowding distance procedure is utilized, so that the diversification of the next generation solutions selected from a Pareto front is assured. However, the 2-OPT local search is implemented on all solutions of the population in the HNSGA-II. Therefore, the assignment of jobs to machines will be constant and only the sequence of jobs will be changed using the 2-OPT local search procedure. We performed the local search procedure on all chromosomes and then added the derived solutions to the population. Afterwards using non-dominated sorting and crowding distance sorting, the chromosomes related to the next generation will be selected.

To validate the performance of algorithms developed in this study, they have been encoded for a bi-objective parallel machine problem in the literature (Berrichi et al. 2009). By comparing the number of Pareto front solutions and the quality of derived solutions, our NSGA-II delivered better performance than NSGA-II of Berrichi et al. (2009). We used MID, RAS, CS, and computer processing time (time) as comparison metrics. After comparing the two algorithms, the following considerable conclusions have been derived:

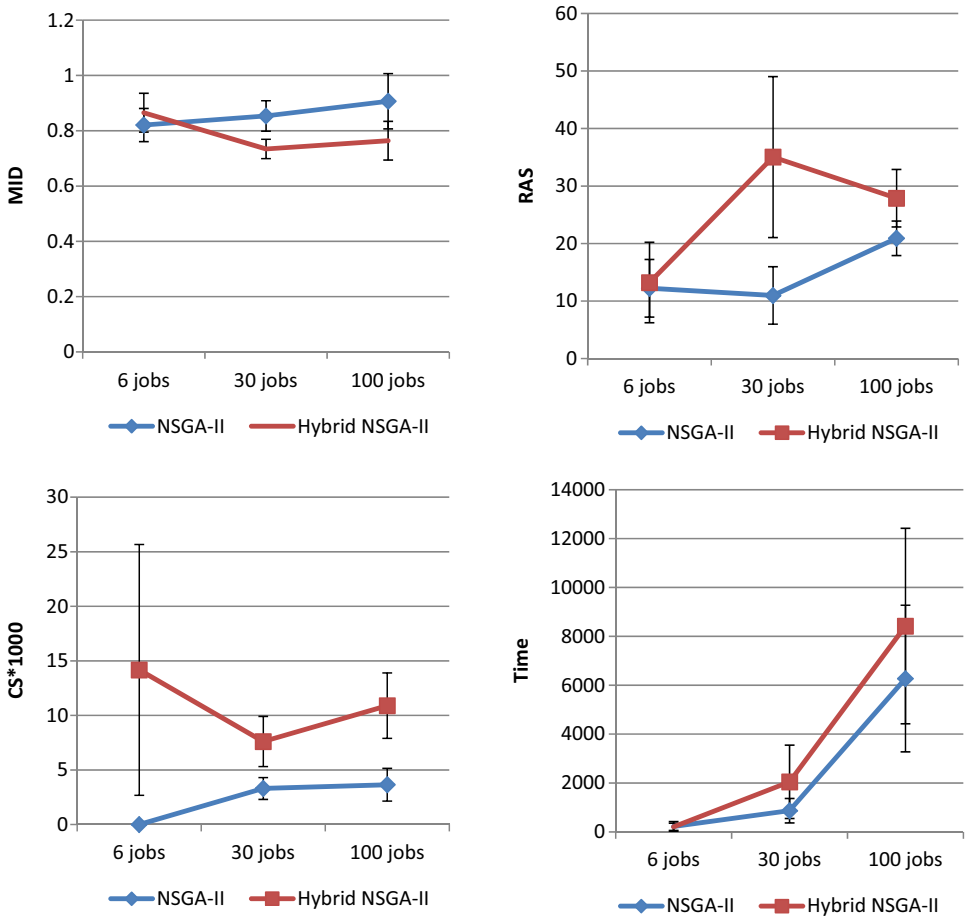

Fig. 8 Confidence intervals according to each comparison metric for both algorithms

1. From CS metric viewpoint, HNSGA-II obviously operates better than NSGA-II.
2. From RAS viewpoint, NSGA-II obviously operates better than HNSGA-II.
3. From MID viewpoint for small-size problems, NSGA-II operates better than HNSGA-II, while for medium- and large-size problems HNSGA-II operates better.

As opportunities for future works, we propose the following studies:

1. HNSGA-II can be implemented to optimize other kinds of problems such as single machine, parallel machines, and problems in job shop environments.
2. Other kinds of local search such as 3-OPT and another kind of 2-OPT can be utilized to hybridize NSGA-II.
3. Some other assumptions can be added to the problem such as considering due dates, random processing times, an transportation times between stages to reflect the real industries' requirements.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interests.

References

Adzapka KP, Adjallah KH, Yalaoui F (2004) On-line maintenance job scheduling and assignment to resources in distributed systems by heuristic-based optimization. J Intell Manuf 15:131–140

Allaoui H, Lamouri S, Artiba A, Aghezzaf E (2008) Simultaneously scheduling n jobs and the preventive maintenance on the two-machine flowshop to minimize the makespan. Int J Prod Econ 112:161–167

Amodeo L, Chen H, El-Hadji A (2007) Multi-objective supply chain optimization: an industrial case study. Appl Evolut Comput 4448:732–741

Angelo-Bello F, Alvarez A, Pacheco J, Martinez I (2011) A heuristic approach for a scheduling problem with periodic maintenance and sequence-dependent setup times. Comput Math Appl 61:797–808

Basseur M (2006) Design of cooperative algorithms for multi-objective optimization: application to flowshop scheduling problems. 4OR 4:255–258

Bechikh S, Belgasmi N, Ben Said L, Ghédira K (2008) PHC-NSGA-II: a novel multi-objective memetic algorithm for continuous optimization. In: 20th IEEE international conference on tools with artificial intelligence, pp 180–189

Behnamian J, Zandieh M, Fatemi Ghomi SMT (2010) A multi-phase covering Pareto-optimum front method to multi-objective parallel machine scheduling. Int J Prod Res 48:4949–4976

Berrichi A, Amodeo L, Yalaoui F, Chatelet E, Mezghiche M (2009) Bi-objective optimization algorithms for joint production and

maintenance scheduling: application to the parallel machine problem. J Intell Manuf 20(4):389–400
Coellom CA, Cortes NC (2002) Solving multi-objective optimization problems using an artificial immune system. In: Evolutionary computation group, Institute Politécnico Nacional No. 2508 Col. San Pedro Zacatenco México
Deb K, Agrawal S, Pratab A, Meyarivan T (2000) A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: KanGAL report 200001, Indian Institute of Technology, Kanpur
Ebeling CE (1997) An introduction to reliability and maintainability engineering. McGraw-Hill, New York
Garey M, Johnson DS (1979) A guide to the theory of NP-completeness. W.H. Freeman and Company, San Francisco
Gaspar-Cunta A, Covas JA (2003) A real-word test problem for EMO algorithms. In: Lecture notes in computer science, LNCS 2632. Springer, Heidelberg, pp 752–766
Gupta JND, Tunc EA (1991) Schedules for a two stage hybrid flowshop with parallel machines at the second stage. Int J Prod Res 29:1489–1502
Hadidi LA, Rahim MA (2015) Reliability for multiple units adopting sequential imperfect maintenance policies. Int J Syst Assur Eng Manag 6(3):103–109
Hadidi LA, Al-Turki UM, Rahim MA (2012a) Joint job scheduling and preventive maintenance on a single machine. Int J Oper Res 13(2):174–184
Hadidi LA, Al-Turki UM, Rahim MA (2012b) Integrated models in production planning and scheduling, maintenance and quality: a review. Int J Ind Syst Eng 10(1):21–50
Hadidi LA, Al-Turki UM, Rahim MA (2015) Practical implications of managerial decisions to integrate production scheduling and maintenance. Int J Syst Assur Eng Manag 6(3):224–230
Hajinejad D, Salmasi N, Mokhtari R (2011) A fast hybrid particle swarm optimization algorithm for flow shop sequence dependent group scheduling problem. Sci Iran 18(3):759–764
Ishibuchi H, Yoshida T, Murata T (2003) Balance between genetic search and local search in memetic algorithms for multi-objective permutation flowshop scheduling. IEEE Trans Evol Comput 7(2):204–223
Kaabi J, Varnier C, Zerhouni N (2002) Heuristics for scheduling maintenance and production on a single machine. In: IEEE conference on systems, man and cybernetics, Hammamet
Kapalanoglu V (2014) Multi-agent based approach for single machine scheduling with sequence-dependent setup times and machine maintenance. Appl Soft Comput 23:165–179
Kurz ME, Askin RG (2004) Scheduling flexible flow lines with sequence-dependent setup times. Eur J Oper Res 159(1):66–82
Landa-Silva JD, Burke EK, Petrovic S (2003) An introduction to multi-objectivemeta-heuristics for scheduling and timetabling. In: Automated scheduling, optimisation and planning research group, School of Computer Science and IT, University of Nottingham, Nottingham
Loukil T, Teghem J, Tuyttens D (2005) Solving multi-objective production scheduling problems using meta-heuristics. Eur J Oper Res 161:42–61
Marett R, Wright M (1996) A comparison of neighborhood search techniques for multi-objective combinatorial problems. Comput Oper Res 23:465–483
Naderi B, Zandieh M, Fatemi Ghomi SMT (2009a) A study on integrating sequence dependent setup time flexible flow lines and preventive maintenance scheduling. J Intell Manuf 20(6):683–694
Naderi B, Zandieh M, Fatemi Ghomi SMT (2009b) Scheduling sequnce-dependent setup time job shops with preventive maintenance. Int J Adv Manuf Technol 43(1–2):170–181
Naderi B, Zandieh M, Aminnayeri M (2011) Incorporating periodic preventive maintenance into flexible flowshop scheduling problems. Appl Soft Comput 11(2):2094–2101
Nilsson C (2003) Heuristics for the traveling salesman problem. Linköping University, Linköping
Rios-Mercado RZ, Bard JF (1998) Computational experience with a branch-and-cut algorithm for flowshop scheduling with setups. Comput Oper Res 25(5):351–366
Shahvari O, Logendran R (2015) Bi-criteria batch scheduling on unrelated-parallel machines. In: Proceedings of the 2015 industrial and systems engineering research conference (ISERC2015), Tennessee
Shahvari O, Logendran R (2016a) Bi-criteria batch scheduling in hybrid flow shop. In: Proceedings of the 2016 industrial and systems engineering research conference (ISERC2016), CA
Shahvari O, Logendran R (2016b) Hybrid flow shop batching and scheduling with a bi-criteria objective. Int J Prod Econ 179:239–258
Shahvari O, Logendran R (2017) An enhanced tabu search algorithm to minimize a bi-criteria objective in batching and scheduling problems on unrelated-parallel machines with desired lower bounds on batch sizes. Comput Oper Res 77:154–176
Shahvari O, Salmasi N, Logendran R (2009) A meta-heuristic algorithm for flexible flow shop sequence dependent group scheduling problem. In: Proceedings of the 2009 International conference on value chain sustainability (ICOVACS 2009), Kentucky
Shahvari O, Salmasi N, Logendran R, Abbasi B (2012) An efficient tabu search algorithm for flexible flow shop sequence-dependent group scheduling problems. Int J Prod Res 50:4237–4254
Van Veldhuizen DA, Lamont GB (1998) Evolutionary computation and convergence to a Pareto front. In: Late breaking papers at the genetic programming conference, Stanford University, Stanford, CA, pp 221–228
Velasco, N., Dejax, P., Guéret, C., and Prins, C. (2006). Genetic algorithm for the bi-objective collection and delivery problem. In: 6th international Francophone conference of modeling and simulation, Mosim'06, Marocco
Vilcot G, Billaut J-C, Esswein C (2006) A genetic algorithm for a bi-criteria flexible job shop scheduling problem. In: IEEE international conference on service systems and service management (ICSSSM'06)
Villemeur A (1991) Reliability, availability, maintainability and safety assessment. Wiley, New York
Yijie S, Gongzhang S (2008) Improved NSGA-II multi-objective genetic algorithm based on hybridization-encouraged mechanism. Chin J Aeronaut 21:540–549
Yulan J, Zuhua J, Wenrui H (2008) Multi-objective integrated optimization research on preventive maintenance planning and production scheduling for a single machine. Int J Adv Manuf Technol 39(9):954–964
Zandieh M, Karimi N (2011) An adaptive multi-population genetic algorithm to solve the multi-objective group scheduling problem in hybrid flexible flowshop with sequence-dependent setup times. J Intell Manuf 22(6):979–989
Zandieh M, Mozaffari E, Gholami M (2010) A robust genetic algorithm for scheduling realistic hybrid flexible flow line problems. J Intell Manuf 21(6):731–743
Zitzler E (1999) Evolutionary algorithms for multi-objective optimization: methods and applications. PhD thesis, Swiss Federal Institute of Technology, Zurich