IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 30, 2020, accepted May 13, 2020, date of publication May 21, 2020, date of current version June 4, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2996496

A Neutrosophic Number-Based Memetic Algorithm for the Integrated Process Planning and Scheduling Problem With Uncertain Processing Times

LIANGLIANG JIN[1], CHAOYONG ZHANG[2], (Member, IEEE), XIAOYU WEN[3], AND GEORGE GERSHOM CHRISTOPHER[4]

[1]Department of Mechanical Engineering, Shaoxing University, Shaoxing 312000, China
[2]School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
[3]Henan Key Laboratory of Intelligent Manufacturing of Mechanical Equipment, Zhengzhou University of Light Industry, Zhengzhou 450002, China
[4]School of Civil Engineering, Shaoxing University, Shaoxing 312000, China

Corresponding author: Liangliang Jin (jll2009@foxmail.com)

This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LQ18E050006, in part by the Funds for the National Natural Science Foundation of China under Grant 51805330, Grant 51575211, Grant 51561125002, and Grant 51905494, in part by the Youth Fund for Humanities and Social Sciences of the Ministry of Education of China under Grant 19YJCZH185, and in part by the 111 Project of China under Grant B16019.

ABSTRACT Process planning and scheduling are two crucial components in a flexible manufacturing system. Lots of novel meta-heuristics have been applied to the integrated process planning and scheduling (IPPS) problem for an efficient utilization of manufacturing resources; nevertheless, the tricky part in real life stems from the uncertainty in processing times. Existing publications regarding IPPS problems mainly focus on cases with nominal or fixed processing times; nevertheless, processing time fluctuations will certainly result in an intolerable deviation between the actual makespan and the nominal one. This research focuses on the IPPS problem with uncertain processing times to hedge against the uncertainty in makespan. The novel neutrosophic numbers are first introduced to model the uncertain processing times. A neutrosophic number based mixed integer linear programming (MILP) model is established; due to the non-deterministic polynomial (NP)-hardness and the complexity in solving the model, a variable neighborhood search (VNS) incorporated memetic algorithm (MA) is then developed to facilitate more robust solutions. In the proposed algorithm, the nominal makespan criterion and the deviation (robustness) criterion have been considered in a weighted sum manner. The well-known Kim's benchmark is adopted to test the performance of the proposed algorithm and different degrees of fluctuations are also defined in experiments. Computational results reveal that the VNS based local search method is powerful in capturing promising solutions; competitive solutions with superior nominal makespan and robustness have been obtained. This research presents a novel perspective or methodology to seek more robust solutions for the uncertain IPPS problem.

INDEX TERMS Neutrosophic numbers, uncertain processing time, integrated process planning and scheduling, robustness, mathematical model, memetic algorithm, variable neighborhood search (VNS), integrated manufacturing systems.

I. INTRODUCTION

Process planning and scheduling are two important functions in manufacturing systems [1]–[5]. Process planning bridges the gap between designing and manufacturing; especially, it specifies the manufacturing sequences for all the operations while scheduling determines the optimal starting and completion times for each operation on corresponding machines such that the maximum completion time (makespan) can be reduced or other criteria can be optimized [6], [7]. Traditionally, the two components perform separately [8]–[10], and due to ignoring the inherent relationship between the two

The associate editor coordinating the review of this manuscript and approving it for publication was Aniruddha Datta.

functions, this paradigm results in many shortcomings that have been well discussed [11], [12], such as low machine utilizations and conflicts of resources [13]. Particularly, the scheduling procedure will be performed after the process planning procedure; therefore, the pregenerated process plans may not be the suitable one in the scheduling module. In order to take the advantage of the flexibilities in both process planning and job shop scheduling, lots of investigations on the IPPS problem have been performed [14]. Except the mathematical programming based approaches, the heuristic rule based solution methods and agent based optimization methods, meta-heuristic algorithms have garnered wide research attentions with relatively fruitful results observed. However, whatever the solution methods adopted, the essence of the optimization lies in how to handle three kinds of flexibilities: operation flexibility (OF, also routing flexibility), sequencing flexibility (SF) and processing flexibility (PF) [15]. Operation flexibility means that there may be more than one available machines can be used in processing an operation. Processing flexibility refers to the possibility in operation selection because there may be more than one alternative operation combinations to produce the same feature for a part [8]. Sequencing flexibility relates to the diversity of operation permutation since for a given set of operations, it may have various permutations provided that the operations satisfy the precedence constraints. The traditional sequential paradigm in performing both the process planning procedure and the scheduling procedure cannot take the advantage of these flexibilities and worse still this paradigm rigidifies these flexibilities.

It was observed that existing publications on the optimization of the IPPS problem either put more emphasis on the optimization method to shorten the makespan by using novel algorithms or focus on multi-objective optimizations by developing various NSGA-II based multi-objective algorithms. Especially, most research papers strength the need of optimizations of the IPPS problem with static processing times, and researches on IPPS problem that consider real world shop floor status have seldom been considered. Nevertheless, due to external or internal disturbances, such as time fluctuations in workpiece loading, the shop floor status (e.g. machining times of operations) changes dynamically [16] and the manufacturing system often subject to many kinds of uncertainties [17]; in such a case, deterministic IPPS problem optimization methods suggested in existing publications can no longer be applied to optimize such processing time varying IPPS problems [18]; otherwise, there would be serious consequences [19]. For instance, due to processing time fluctuations caused by previous operations processing procedures, the actual starting time of an operation on a certain machine usually deviates from the the predefined value (the starting time in the optimal schedule with respect to deterministic models or algorithms, also the nominal starting time) and this may further cause larger starting time deviations in processing following operations because in such a case subsequent operations will have to be put off. Consequently, the actual scheduling scheme in the shop floor may totally different with the predefined ''optimal'' deterministic scheduling scheme, and the staff in shop floor will face the risk where the so-called optimal schedule obtained using deterministic model become totally useless [20].

In this research, we mainly focus on the IPPS problem with uncertain processing times because the processing times of operations in the real world usually fall within a certain range instead of a deterministic value and more importantly this processing time uncertainty is quite common during manufacturing procedures compared with other unexpected situations. Therefore, the IPPS problem with uncertain processing times is urgent to be addressed to make the optimal scheduling scheme become more practical. Nevertheless, few researches regarding such problems have been conducted by far due to the complexity of the problem itself as well as the uncertain processing time modelling. To our knowledge, similar to the off-line and the online paradigms, the solution methods to tackle uncertain scheduling problems can be classified into two categories: the reactive scheduling method and the proactive (or preventive) scheduling method [21]. For the reactive scheduling method, it regenerates first an optimal scheduling scheme for the jobs in the shop floor and whenever unexpected events occur, e.g. there is a large time deviation between the actual completion time and the nominal value, a new and updated scheduling scheme will be generated timely for the remainder operations. Clearly, it works in an online manner and the critical failing is that the global optimality is not ensured because a part of operations have already been processed. For the proactive scheduling method, it considers processing time fluctuations in advance and the resultant scheduling scheme has a certain immunity to processing time fluctuations and therefore this off-line paradigm is capable to hedge against time fluctuation of makespan.

For convenience, many researchers deemed uncertain processing times as variables that subject to a certain distribution, e.g. Gaussian distribution; other researchers treat uncertain processing times as fuzzy numbers that subject to certain membership functions [17]. However, it is usually quite difficulty to identify the probability distributions or the membership functions that uncertain processing times follow. Even though the probability distribution can be determined, it usually requires lots of prior knowledge and data. Meanwhile, in most cases, the actual processing time of an operation falls within a certain range, e.g. $[a,\ b]$, and it is quite easy to detect the upper and lower bounds a and b of the interval; this brings convenience to the modelling the uncertain processing times as well as the optimization procedure. In this study, the neutrosophic numbers are introduced in solving this uncertain IPPS problem for the first time. Realizing the difficulty in conveying people's thinking, Smarandache in 2008 first proposed neutrosophic numbers [22], [23], which consists of a determinate part and an indeterminate parts. the main idea of neutrosophic numbers corresponds to real life situations: usually, people can only tell a certain range which a value will fall within instead of a determined value. Typically, a

neutrosophic number can be expressed as $N = d + uI,\ \ d, u \in \mathbb{R}$, where d stands for the determinate part and I, uI mean the indeterminacy and the indeterminate parts respectively. Particularly, a neutrosophic number will degenerate to a real number provided that u equals 0; in the worst case where $d = 0$, however, a neutrosophic number N can be expressed as uI. Clearly, neutrosophic numbers are very suitable in indeterminate information expressing [24]; the background of neutrosophic numbers tallies with the situations of uncertain processing times in machining procedures: because one only knows the low bounds, the upper bounds and the fluctuations of processing times, the determinate part and the indeterminate part exactly corresponds to the low bounds and the fluctuations of processing times respectively. In fact, applications of neutrosophic numbers have been attracting researchers' attentions; by far, applications of neutrosophic numbers have been seen in (group) decision supporting [24]–[28], truss structure optimal designing with uncertain parameters [29], steam turbine fault diagnosing [30]. For instance, Ye *et al.* developed a neutrosophic number optimization model for truss structure design problems and their research provides a novel and effective way for truss structures designing under indeterminate environments [29]. Chen and Ye [25] developed a neutrosophic number based decision making tool to select clay bricks; the projection method of neutrosophic numbers is adopted to obtain the projection measure between two alternatives. In this paper, we try to take the advantage of neutrosophic numbers in expressing indeterminate information or ambiguity of people's cognition; uncertain processing times in this paper are modelled as neutrosophic numbers and corresponding optimization method will be developed to capture a more robust scheduling scheme. Since the plain IPPS problem is a complex NP-hard problem, the uncertain processing times make the problem become more complicated; this research therefore adopts a variable neighborhood search (VNS) based memetic hybrid algorithm to tackle the problem.

In general, due to the swarm intelligence in genetic algorithm (GA), it is very suitable to solve large scale and complex (job shop) scheduling problems; for example Li *et al.* adopted the GA in IPPS optimization [9]. Compared with other meta-heuristic algorithms, GA has the ability to retain the diversity of individuals to avoid being trapped into local optima. Nevertheless, even though the GA has the ability to explore new solutions by using its swarm intelligence, sometimes many solutions (or individuals) can still go into local optima because the fitness landscape or the solution space is quite complicated. In such cases, the plain GA cannot be applied directly to the problem. Based on our observations, the best way to deal with such a situation is employing problem specific heuristic methods to further improve the solution quality. In relatively early researches, effective problem oriented heuristic search techniques have been frequently investigated [31]–[34]; for instance, Zhang *et al.* reported a successful application of neighborhood structure based heuristic optimization technique for the flexible job shop scheduling problem [35], [36]. It comes to our mind that the optimization quality will be improved if both the meta-heuristic algorithm and the problem-oriented heuristic method are employed. In other words, the hybrid of both the constructive heuristics as well as the improvement heuristics should be included and hybridized [37], and thus, the memetic algorithm (MA) [38] can be adopted in this research. In this research, the N5 neighborhood structure [32] as well as the neighborhood structure proposed by mastrolilli [33] (abbreviated as neighborhood Nm in this paper) are applied in the VNS local search method to intensify the search ability of the MA. The N5 neighborhood structure has high-cost performance because only swapping the first and the last two critical operations are needed; the swapping process can be a tiny perturbation on an individual but may receive appreciable improvements in solution quality. The neighborhood Nm is developed based on the critical path, which determines the makespan of a schedule scheme. It tries to remove the operations on the the critical path to another alternative machine such that the original critical path will be broken and the length of the new critical path may be shortened. Instead of following a trajectory of the optimization of a single solution, the VNS search technique can explore neighborhoods of current incumbent solutions and perform systematic jumps between neighborhoods to achieve more improvements [39]. Especially in scheduling problem optimizations, when a solution cannot be further optimized by a single local search technique, the VNS local search method provides a new perspective for further improvements since the solution may be improved by other neighborhood jumps, and this is the reason why the VNS local search method is applied in this research. By far, the successful applications of the VNS local search algorithm in solving complex problems have been reported across many industrial sectors, such as flexible job shop scheduling [40], bicycle sharing system balancing [41], and the vehicle-routing problem with time windows [42], etc.

This research tries to develop an effective optimization approach to capture more promising and robust scheduling schemes for the IPPS instances. In this research, neutrosophic numbers enable the modeling of the uncertain processing times and further the neutrosophic number based optimization method is developed. One of the merit of this research lies in that it provides a novel perspective or a general optimization framework for the the IPPS problem which is inevitably contaminated with uncertain parameters (processing times). The novelty of this research can be concluded as follows:

- This research presents a novel perspective to facilitate robustness improvement as well as makespan reduction in uncertain IPPS problem optimization. Especially, neutrosophic numbers are first introduced in meta-heuristic algorithms to model uncertain processing times. Such optimization method has not been investigated in existing research publications.
- Due to the complexity in solving the MILP model as well as the problem itself, we develop an effective memetic

algorithm where the problem specific N5 and Nm neighborhood structures in VNS local search method are considered to obtain more promising results. Both the robustness criterion and the makespan criterion have been optimized in a weighted sum manner because in most cases the deterioration of the makespan criterion will occur if only the robustness is considered in the algorithm.

- A novel neutrosophic number based MILP model is proposed to accommodate processing time fluctuations.

The remainder of the paper is organized as follows. We briefly review some relative articles concerning the uncertain IPPS problem or uncertain scheduling problems in next section. After this, some basic concepts and arithmetic operations related to neutrosophic numbers together with the neutrosophic number based MILP model will be proposed. Section IV demonstrates the details as well as the work flow of the proposed memetic algorithm. Section V performs the experimental study, where the well-known Kim's benchmark [13] is adopted to test the proposed algorithm, and moreover, we compare the resultant solutions and analyze the results. The conclusion as well as some future research directions will be presented in the last section to finalize this paper.

II. RELATED LITERATURE

The IPPS problem has been investigated extensively several years before [5], [13], [43]–[47]. Traditional research on this problem mainly considers the makespan reduction with deterministic processing times using novel algorithms or keeps an eye on multi-objective optimization methods [48]; especially, the variants of non-dominated sorting techniques. For example, Petrovic *et al.* proposed a chaotic PSO algorithm to address the deterministic IPPS problem; the machine utilization criterion and the mean flow time criterion have also been considered [49]. In recent years, the research attention of the IPPS problem has been extended and more practical or real life situation oriented factors have been included in optimization of the IPPS problem [50]. For instance, Seker *et al.* developed a hybrid heuristic model which combines both a clustered chromosome structure based GA and fuzzy neural network (FNN) for the IPPS problem to accommodate the changing conditions including order cancelations, uncertain due date, machine breakdowns, etc., [51]. In our opinion, hedge against the processing time uncertainty and keep the robustness of a schedule scheme is of top priority since the fluctuations of starting times and ending times of operations will disturb the predefined optimal schedule scheme and subsequent manufacturing procedures will also be put off. In such a case, the staff in the shop floor are exposed to the risk that the so-called ''optimal'' schedule obtained by a deterministic approach performs poorly in real world manufacturing processes [20]. Although periodic and event-driven rescheduling of unprocessed operations can to a certain extent improve the worsening production situations, this rescheduling for partial operations cannot ensure the global optimality [14], and usually, the makespan after rescheduling is longer than that of the optimal scheduling scheme. In the following, we mainly discuss the literature that deal with uncertain parameters in scheduling.

The first kind of methods to deal with uncertain processing time in scheduling is the chance constrained programming (CCP) approaches; this approach has first been considered in solving scheduling problems with uncertain parameters according to existing research publications [52]–[55]. By this method, a MILP model that relates to the indeterministic parameters can be easily converted into a deterministic model and hence it can be solved using commercial solvers. Elyasi and Salmasi give an example on the application of such method [55]; the single machine scheduling and the flow shop scheduling problems are studied and the processing times are assumed to be random variables. Recently, Liu *et al.* performed a study on the stochastic flow shop scheduling problem and a new distributionally robust chance constrained model is proposed [56]. Other than scheduling problems, chance constrained programming based approaches have also been applied in other problems, such as fast-charging station planning [57], water resources planning and pollution controlling [58], disassembly sequence planning [59], etc. However, for medium or large sized job shop scheduling problems the computing time in solving resultant deterministic mathematical models generated by the CCP method is quite long and this is due to the nature of NP-hardness.

The fuzzy number based methods, belonging to the other type of approaches in addressing indeterministic scheduling problems, have also been considered by researchers. Due to the nature of fuzzy numbers, scheduling problems with fuzzy processing times can also be solved by meta-heuristics and this brings convenience to the optimization of such uncertain NP-hard problems; by far, meta-heuristics are deemed as one of the most effective and efficient approaches in dealing NP-hard problems. Wang *et al.* presented a hybrid artificial bee colony (HABC) algorithm for the fuzzy flexible job shop scheduling problem [60]; the indeterministic processing times are modeled with triangular fuzzy numbers and corresponding encoding and decoding techniques were also developed. Lei developed an efficient decomposition-integration genetic algorithm (DIGA) to minimize the maximum fuzzy completion time [61]; the uncertain processing times are also mapped into triangular fuzzy numbers. They further proposed a new max operation of two fuzzy numbers by approximation to replace the Sakawa criterion. In fact, Sakawa and Mori are the pioneers who first modelling uncertain processing time with fuzzy numbers [62]: when they first applied triangular fuzzy numbers in solving the job shop scheduling problem with uncertain processing times and duedates the max operator of fuzzy numbers was approximated by a certain formula because the result after the max operator may not become a triangular fuzzy number, and this is the flaw of the fuzzy number based solution approach in solving uncertain scheduling problems. Later, Wang *et al.* performed a similar research for the flexible job shop scheduling problem

with fuzzy processing time by using an effective estimation of distribution algorithm (EDA) [63]; again, the uncertain processing times were modeled with triangular fuzzy numbers. In recent years, Gao *et al.* considered triangular fuzzy processing times and new job arrivals simultaneously in a flexible job shop and a two-stage artificial bee colony (TABC) algorithm with several improvements is adopted to minimize the fuzzy makespan [64]. Jamrus *et al.* presented a hybrid particle swarm optimization algorithm where genetic operators are applied for flexible job shop scheduling problem in semiconductor industry [65]; the uncertain processing times are also expressed using triangular fuzzy numbers. Other than triangular fuzzy numbers, other fuzzy numbers, such as trapezoidal fuzzy numbers, are seldom used to represent uncertain processing times. However, in some cases, one don't know actual processing times follow which probability distribution and only the rough low bounds and the upper bounds of processing times are available in most cases; therefore, actual processing time in this case cannot be presented as triangular fuzzy numbers unless the membership function obtained from historical data is available. This is the other flaw of applying fuzzy numbers for uncertain scheduling problems. Different with traditional fuzzy numbers, only the upper and the low bounds are required in neutrosophic numbers; this brings convenience in problem optimization.

Except the two approaches mentioned above, other methods have also been reported. Haddadzade *et al.* considered the stochastic process time IPPS problem, and a two stage method is proposed: the CAPP system first generates all the process plans and four near-optimal process plans are selected later to build a robust scheduling scheme [6]; however, their method splits the process planning module and the scheduling module and the selected four process plans may not the best one in the scheduling module. Horng *et al.* suggested a metaheuristic algorithm called evolutionary strategy in ordinal optimization (ESOO) for the stochastic job shop scheduling problem to minimize the sum of storage expenses and tardiness penalties [66]; the whole algorithm was divided into two stages: the rough stage and the softening procedure. Wang *et al.* developed a novel decomposition-based holonic approach (DBHA) to minimize the makespan for the flexible flow shop scheduling problem with stochastic processing times [67]. The Petri net based method is also covered. Liu *et al.* presented a Petri net based model for an emergency response process constrained by resources and uncertain durations [68]. In this research, we adopt neutrosophic numbers to present uncertain processing times and a novel MILP model with VNS based solution approach is proposed to hedge against the uncertainty and improve the robustness of the resultant scheduling scheme.

III. MATHEMATICAL MODELLING

A. PROBLEM DESCRIPTION

The IPPS problem can be defined as [69]: Given a set of n parts (jobs) to be processed on m machines with operations

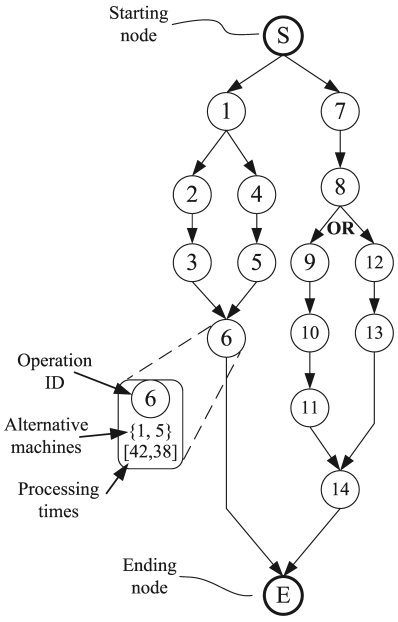


FIGURE 1. A network graph.

including alternative manufacturing resources, select the suitable manufacturing resources and sequence the operations so as to determine a schedule in which the precedence constraints among operations can be satisfied and the corresponding objectives, e.g. the robustness and the makespan criteria, can be optimized. In this research, the actual processing time of each operation on each available machine will fall within a certain range and can be presented as a neutrosophic number.

Usually, jobs to be processed in the IPPS problem are represented by network graphs shown in Figure 1. The starting node and the ending node are dummy nodes respectively; they mean the start and the completion in processing a job. Other nodes in figure 1 are operation nodes and OR nodes (marked with "OR"(s)). In each operation node, Operation 6 in Figure 1 for example, the operation ID and the available machines which are eligible to process this operation with corresponding nominal processing times are provided. Therefore, Operation 6 in the figure can be provided by Machines 1 and 5 with corresponding nominal processing times 42 and 38 respectively. If an arrow comes from node A to node B, it implies that operation A should be processed before operation B directly or indirectly. An operation path that begins at an OR node and ends as it merges with other paths is called an OR link path [15], and only the operation nodes in exactly one of the two operation paths will be visited. In the case of Figure 1, the two operation paths are Operations 9, 10, 11 and Operations 12, 13. It should be noted that some bifurcations are not caused by "OR" nodes, and all the operations in link paths are needed to be visited; for example, Operations 2, 3 and Operations 4, 5 should all be visited. A job in IPPS problem may have a large number of feasible process plans depending on different operation permutations because any

operation permutations are feasible as long as the operations satisfy the precedence relationships expressed by the one-way arrows in the network graph. In many cases, the precedence relationships of some operations are not fixed (e.g. operations 5 and 9 in Figure 1) because there are no arrows that link these operations. Thus, Operation O_i may not appear between Operations O_{i-1} and O_{i+1}, and this is quite different from the case in (flexible) job shop scheduling problems. Two possible process plans in Figure 1 can be ($7 \rightarrow 1 \rightarrow 4 \rightarrow 8 \rightarrow 12 \rightarrow 13 \rightarrow 2 \rightarrow 5 \rightarrow 3 \rightarrow 6 \rightarrow 14$) and ($1 \rightarrow 7 \rightarrow 2 \rightarrow 8 \rightarrow 4 \rightarrow 9 \rightarrow 5 \rightarrow 10 \rightarrow 3 \rightarrow 11 \rightarrow 14 \rightarrow 6$).

The network graph reflects the three kinds of flexibilities as mentioned in Section I. Operation flexibility relates to the case where most operations in the network graph have more than one available machines. Sequencing flexibilities means one job can have many alternative operation permutations (process plans) depending on the precedence relationships expressed by the arrows between operations in a network graph. The processing flexibility can also be expressed by the network graph: only the operations belonging to one of the OR link paths will be selected. Therefore, the essence of the IPPS optimization is to make a full use of the three flexibilities to make a smaller makespan value and obtain a more robust scheduling scheme.

B. PRELIMINARIES

Neutrosophic number operators are required to be defined. Three neutrosophic number operators, e.g. the addition operator, the maximization operator and the ranking operator, are included in this research. The addition operator is used to determine the completion time of an operation; the maximization operator is adopted to calculate the starting time of an operation, and the the ranking operator is included to compare two neutrosophic numbers so as to determine the maximum complection time (makespan).

Smarandache first proposed the concept of a neutrosophic number [22], [23]: $N = a + bI$, where a and b are two real numbers while I is the indeterminacy which satisfies $I^2 = I$ and $0 \cdot I = 0$. Thus, a neutrosophic number $N = 2.4 + 0.4I, I \in [0,\ 0.5]$ means that the determinacy is 2.4 and the indeterminacy can vary in the interval $[0,\ 0.2]$; it is equivalent to $N \in [2.4,\ 2.6]$. Further, it can also be expressed as $N = 2.4 + 0.2I,\ I \in [0,\ 1]$. Let $N_1 = a_1 + b_1 I$ and $N_2 = a_2 + b_2 I$ be two neutrosophic numbers, Smarandache gives the following operators [22], [23]:

$$
\begin{aligned}
&1. N_1 + N_2 = a_1 + a_2 + (b_1 + b_2)I;\\
&2. N_1 - N_2 = a_1 - a_2 + (b_1 - b_2)I;\\
&3. N_1 \times N_2 = a_1 a_2 + (a_1 b_2 + b_1 a_2 + b_1 b_2)I;\\
&4. \frac{N_1}{N_2} = \frac{a_1}{a_2} + \frac{a_2 b_1 - a_1 b_2}{a_2(a_2 + b_2)}I,\ a_2 \neq 0,\ a_2 + b_2 \neq 0. \qquad (1)
\end{aligned}
$$

Therefore, by the addition operator, the sum of two neutrosophic numbers, $N_1 = 2.4 + 0.2I$ and $N_2 = 0.8 + 0.4I,\ I \in [0,\ 1]$, is $3.2 + 0.6I$.

The ranking method is used to compare two neutrosophic numbers; based on the related investigation [70], Ye in 2016 proposed the neutrosophic number oriented ranking method [26]: let $N_i = a_i + b_i I,\ I \in \left[\beta^-,\ \beta^+\right],\ i = 1, 2, 3, \cdots, n$ be an any neutrosophic number with $a_i,\ b_i,\ \beta^-,\ \beta^+ \in \mathbb{R}$. The possibility degree can be defined as follows to compare two neutrosophic numbers $N_i = a_i + b_i I$ and $N_j = a_j + b_j I$ as (2), shown at the bottom of this page.

To compare n neutrosophic numbers $N_i = a_i + b_i I,\ i = 1, 2, 3, \cdots n$, one should perform pairwise comparisons between any two neutrosophic numbers using Eq.2: $P_{ij} = P(N_i \geq N_j)$; the matrix of possibility degrees, $P = (P_{ij})_{n \times n}$, can then be constructed [26], and clearly, the elements of the matrix follow that $P_{ij} \geq 0,\ P_{ij} + P_{ji} = 1$, and $P_{ii} = 0.5$. The ranking order of the $i-$th neutrosophic number can be calculated with Eq.3 [26].

$$
r_i = \frac{\left(\sum_{j=1}^{n} P_{ij} + \frac{n}{2} - 1\right)}{n\,(n-1)} \qquad (3)
$$

The maximization operator used in this research is quite easy. An operation can be started to process only when its job predecessor (JP) and the machine predecessor (MP) are completed, whose completion times are also neutrosophic numbers. Suppose there are two neutrosophic numbers representing the completion times of JP and MP, e.g. $N_1 = a_1 + b_1 I,\ N_2 = a_2 + b_2 I,\ I \in \left[\beta^-,\ \beta^+\right]$; the starting time of the current operation can be defined as $\left[\max\{a_1 + b_1\beta^-, a_2 + b_2\beta^-\}, \max\{a_1 + b_1\beta^+, a_2 + b_2\beta^+\}\right]$, and it is also a neutrosophic number after further transformations.

C. MODEL BUILDING

Based on our observation, neutrosophic number based mathematical models have not been considered. In this section, we try to establish a neutrosophic number based MILP model to reduce the total completion time, which is also a nutrosophic number. According to our previous research, The MILP models of the IPPS problem can be divided into two categories: Type-1 models [71] and Type-2 models [15]. The essential difference of the two kinds of models is that all the possible process plans should be generated when Type-1 models are applied. In general, MILP models of scheduling problems can be expressed as:

$$
\begin{aligned}
&\min_{\boldsymbol{x},\boldsymbol{y}} / \max \quad \boldsymbol{c}^T\boldsymbol{x}\\
&s.t.\ \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} \leq \boldsymbol{p}\\
&\qquad \boldsymbol{E}\boldsymbol{x} + \boldsymbol{F}\boldsymbol{y} = \boldsymbol{e}
\end{aligned}
$$

$$
P_{ij} = P(N_i \geq N_j) = \max\left\{1 - \max\left(\frac{(a_j + b_j\beta^+) - (a_i + b_i\beta^-)}{(a_i + b_i\beta^+) - (a_i + b_i\beta^-) + (a_j + b_j\beta^+) - (a_j + b_j\beta^-)}, 0\right), 0\right\} \qquad (2)
$$

$$\begin{aligned} &\boldsymbol{x}_{\min} \leq \boldsymbol{x} \leq \boldsymbol{x}_{\max} \\ &y = 0,\ 1 \end{aligned} \tag{4}$$

Further, as the case of a complex that can be divided into real and imaginary parts, a neutrosophic number also consists of two parts: the determinacy and the indeterminacy: $z = \bar{z} + \widetilde{z}I$. By applying the multiplication rule in Formula 1 and regarding a real number a as a special neutrosophic number $z = a + 0I$, the model in Formula 4 can be reformed by two submodels:

$$\begin{aligned} &\min_{\bar{x},y} / \max_{\bar{x},y} \bar{\boldsymbol{c}}^T \bar{\boldsymbol{x}} \\ &s.t.\ \bar{A}\bar{\boldsymbol{x}} + \bar{\boldsymbol{B}}\boldsymbol{y} \leq \bar{\boldsymbol{p}} \\ &\quad \bar{E}\bar{x} + \bar{\boldsymbol{F}}\boldsymbol{y} = \bar{\boldsymbol{e}} \\ &\quad \bar{\boldsymbol{x}}_{\min} \leq \bar{\boldsymbol{x}} \leq \bar{\boldsymbol{x}}_{\max} \\ &\quad y = 0, 1 \end{aligned} \tag{5}$$

$$\begin{aligned} &\min_{\widetilde{x},y} / \max_{\widetilde{x},y} \left(\bar{\boldsymbol{c}}^T \widetilde{\boldsymbol{x}} + \widetilde{\boldsymbol{c}}^T \bar{\boldsymbol{x}} + \widetilde{\boldsymbol{c}}^T \widetilde{\boldsymbol{x}}\right) I \\ &s.t.\ \bar{A}\widetilde{\boldsymbol{x}}I + \widetilde{A}\bar{\boldsymbol{x}}I + \widetilde{A}\widetilde{\boldsymbol{x}}I + \widetilde{\boldsymbol{B}}\boldsymbol{y}I \leq \widetilde{\boldsymbol{p}}I \\ &\quad \bar{E}\widetilde{\boldsymbol{x}}I + \widetilde{E}\bar{\boldsymbol{x}}I + \widetilde{E}\widetilde{\boldsymbol{x}}I + \widetilde{\boldsymbol{F}}\boldsymbol{y}I = \widetilde{\boldsymbol{e}}I \\ &\quad \widetilde{\boldsymbol{x}}_{\min} I \leq \widetilde{\boldsymbol{x}}I \leq \widetilde{\boldsymbol{x}}_{\max} I \\ &\quad y = 0, 1 \end{aligned} \tag{6}$$

Formula 5 is totally equivalent to the plain MILP models such as the one in Formula 4; nevertheless, since variables $\bar{\boldsymbol{x}}$ and $\boldsymbol{y}$ both appear in the two submodels, they should be optimized simultaneously. After eliminating the indeterminacy I in the second submodel, both of the models can be combined as shown in Formula 7.

$$\begin{aligned} &\min_{\widetilde{x},\bar{x},y} / \max \quad \left(\bar{\boldsymbol{c}}^T \widetilde{\boldsymbol{x}} + \widetilde{\boldsymbol{c}}^T \bar{\boldsymbol{x}} + \widetilde{\boldsymbol{c}}^T \widetilde{\boldsymbol{x}}\right) + \bar{\boldsymbol{c}}^T \bar{\boldsymbol{x}} \\ &s.t.\ \bar{A}\bar{\boldsymbol{x}} + \bar{\boldsymbol{B}}\boldsymbol{y} \leq \bar{\boldsymbol{p}} \\ &\quad \bar{E}\bar{\boldsymbol{x}} + \bar{\boldsymbol{F}}\boldsymbol{y} = \bar{\boldsymbol{e}} \\ &\quad \bar{A}\widetilde{\boldsymbol{x}} + \widetilde{A}\bar{\boldsymbol{x}} + \widetilde{A}\widetilde{\boldsymbol{x}} + \widetilde{\boldsymbol{B}}\boldsymbol{y} \leq \widetilde{\boldsymbol{p}} \\ &\quad \bar{E}\widetilde{\boldsymbol{x}} + \widetilde{E}\bar{\boldsymbol{x}} + \widetilde{E}\widetilde{\boldsymbol{x}} + \widetilde{\boldsymbol{F}}\boldsymbol{y} = \widetilde{\boldsymbol{e}} \\ &\quad \bar{\boldsymbol{x}}_{\min} \leq \bar{\boldsymbol{x}} \leq \bar{\boldsymbol{x}}_{\max} \\ &\quad \widetilde{\boldsymbol{x}}_{\min} \leq \widetilde{\boldsymbol{x}} \leq \widetilde{\boldsymbol{x}}_{\max} \\ &\quad y = 0,\ 1 \end{aligned} \tag{7}$$

After solved by corresponding solvers, the optimal solution $\bar{\boldsymbol{c}}^T \bar{\boldsymbol{x}} + \left(\bar{\boldsymbol{c}}^T \widetilde{\boldsymbol{x}} + \widetilde{\boldsymbol{c}}^T \bar{\boldsymbol{x}} + \widetilde{\boldsymbol{c}}^T \widetilde{\boldsymbol{x}}\right) I$ is obtained. Based on our prevoiusly proposed Type-2 MILP model [15], the neutrosophic number based MILP model for the uncertain IPPS problem can thus be established. Some assumptions are made: 1)Job preemptions are not allowed; each machine can process one job at a time and each job can only be processed by one machine at any time. 2)Jobs are released at time zero. 3)The set up times as well as transmission times are merged into processing times. 4)a job will be immediately transferred to the next machine once it is finished on current machine. Corresponding sets, subscriptions, parameters, etc. are first introduced as follows.

Subscripts & Notations

i, i'	jobs, $1 \leq i, i' \leq \lvert n \rvert$,
j, j'	operations, $1 \leq j, j' \leq \lvert n_i \rvert$,
k, k'	machines,
h	combinations,
O_{ij}	the j-th operation of job i,
O_{ihj}	the j-th operation of job i using the h-th process plan combination of that job.

Sets & Parameters

A	a very large positive integer,
$\bar{p}_{ijk} + \widetilde{p}_{ijk}I$	the neutrosohic processing time of O_{ij} on machine k; $\bar{p}_{ijk}$ and $\widetilde{p}_{ijk}$ are the determinacy and the indeterminacy parts of a neutrosophic number respectively,
R_{ih}	the set that contains the operations belonging to the h-th combination of job i,
$V_{ijj'}$	1, if O_{ij} is to be processed before $O_{ij'}$ represented directly by the network graph (if there is an arrow coming from node j to node j' in the network graph of job i, then $V_{ijj'} = 1$); 0, otherwise; This parameter can only express precedence relationships for partial operations.
K_i	the set of combinations of job i,
n	the set of all the jobs,
n_i	the set of all the operations in a network graph of job i,
M_{ij}	the set of available machines for O_{ij},
POS_{ij}	the pre-ordered set of O_{ij}; it contains all the operations that should appear before O_{ij},
BOS_{ij}	the back-ordered set of O_{ij}; it contains all the operations that should appear after O_{ij},
$Q_{ijj'}$	1, O_{ij} should be processed directly or indirectly before $O_{ij'}$; 0, otherwise.

Because there exist operations with no precedence relationships, the parameter $V_{ijj'}$ is not enough to distinguish and filtrate the operations that have no precedence relationships. The pre-ordered set (POS) and the back-ordered set (BOS) as well as parameters $Q_{ijj'}$ are constructed based on the parameters $V_{ijj'}$ using the algorithm given in Algorithm 1 [15].

Variables

$\bar{C}_{\max} + \widetilde{C}_{\max}I$	makespan; $\bar{C}_{\max}$ and $\widetilde{C}_{\max}$ are the determinacy and the indeterminacy parts,
Y_{ih}	1, if the h-th combination of job i is selected; 0, otherwise,
$Z_{ijj'}$	1, if operation O_{ij} is processed directly or indirectly before $O_{ij'}$; 0, otherwise,
X_{ihjk}	1, if operation O_{ihj} is processed on machine k; 0, otherwise,
$\bar{C}_{ihj} + \widetilde{C}_{ihj}I$	the complection time of O_{ihj}; $\bar{C}_{ihj}$ and $\widetilde{C}_{ihj}$ are the determinacy and the indeterminacy parts.

Objective(s)

The objective is to minimize the maximum neutrosophic completion time; from Formula 7, the objective contains two

Algorithm 1 Determine Parameters $Q_{ij,j'}$

```
for j := 1 to |n_i| do
  for j' := 1 to |n_i| do
    if (j ≠ j')and(V_{ij,j'} = 1) then
      Q_{ij,j'} := 1
    end if
  end for
end for
for j := 1 to |n_i| do
  for j' := 1 to |n_i| do
    for j'' := 1 to |n_i| do
      if (j ≠ j')and(j' ≠ j'')and(j ≠ j'')and(Q_{ij,j'} = 1)and(Q_{ij',j''} = 1) then
        Q_{ij,j''} := 1
      end if
    end for
  end for
end for
```

parts ($\bar{c}^T\bar{x}$ and $\left(\bar{c}^T\widetilde{x}+\widetilde{c}^T\bar{x}+\widetilde{c}^T\widetilde{x}\right)$), and therefore, both the nominal makespan value as well as the fluctuation of the makespan will be minimized.

$$\min \quad \bar{C}_{\max}+\widetilde{C}_{\max} \tag{8}$$

Constraints

Constraint set 9 means that exactly one operation combination should be selected for each job. The ''operation combination'' contains the operation nodes performing which a part (job) can be complete and precedence relationships between operations are not considered in the combination; it can be generated by judging the OR link paths in a network graph and picking several operations in a network graph. For instance, there are two operation combinations (operations 1-11,14) and (operations 1-8,12-14) in the network graph in Figure 1.

$$\sum_{h\in K_i} Y_{ih}=1, \quad \forall i\in n \tag{9}$$

Each operation in the selected operation combination will be allocated to one of its alternative machines, and unselected operations should be eliminated. Constraint set 10 undertakes this task; if a combination is not selected, the variable Y_{ih} is set to 0 and hence operations in this combination will not be assigned to any machines.

$$\sum_{k\in M_{ij}} X_{ihjk}=Y_{ih}, \quad \forall i,\ h\in K_i,\ j\in R_{ih} \tag{10}$$

For the unselected operations that belong to the h-th combination, corresponding completion times are set to zero; otherwise, completion times of the selected operations are determined by other constraints.

$$A\cdot Y_{ih}\geq \bar{C}_{ihj}, \quad \forall i,\ h\in K_i,\ j\in R_{ih} \tag{11}$$

$$A\cdot Y_{ih}\geq \widetilde{C}_{ihj}, \quad \forall i,\ h\in K_i,\ j\in R_{ih} \tag{12}$$

For two operations that belong to the same jobs and have a direct precedence relationship ($V_{ijj'}=1$ or $V_{ij'j}=1$), the completion times of the two operations can be determined by the constraints as follows.

$$\bar{C}_{ihj'}\geq \bar{C}_{ihj}+\sum_{k'\in M_{ij'}} X_{ihj'k'}\bar{p}_{ij'k'}, \quad \forall i,\ h\in K_i,\quad j,j'\in R_{ih},\ j\neq j',\ V_{ijj'}=1 \tag{13}$$

$$\widetilde{C}_{ihj'}\geq \widetilde{C}_{ihj}+\sum_{k'\in M_{ij'}} X_{ihj'k'}\widetilde{p}_{ij'k'}, \quad \forall i,\ h\in K_i,\quad j,j'\in R_{ih},\ j\neq j',\ V_{ijj'}=1 \tag{14}$$

As mentioned above, some operations in an operation combination have no precedence relationships; and the set POS_{ij}, BOS_{ij} as well as parameters $Q_{ijj'}$ are constructed to eliminate the operations that have precedence relationships. Suppose there are two operations O_{ij} and $O_{ij'}$; if $O_{ij'}$ appears in the set POS_{ij} or BOS_{ij}, then it means that the two operation have a precedence relationship; otherwise, constraints should be introduced to ensure there is exactly one precedence relationship between two operations that initially have no precedence relationship. For example, operations 1 and 4 in Figure 1 will certainly appear in POS of operation 5, and operation 6 will appear in BOS of operation 5; we have $Q_{i1,5}=1$, $Q_{i4,5}=1$, $Q_{i5,6}=1$. However, operation 9 has no precedence relationships with operation 5, and it will never appear in POS or BOS of operation 5; both $Q_{i9,5}$ and $Q_{i5,9}$ equal zero. For such operations, the following constraint set is introduced.

$$\begin{aligned} &Z_{ijj'}+Z_{ij'j}=1, \quad \forall i,\ j,j'\in n_i,\ j\neq j' \\ &Q_{ijj'}+Q_{ij'j}=0,\ \left(orj,j'\notin POS_{ij},\ j,j'\notin BOS_{ij}\right) \end{aligned} \tag{15}$$

Following this, the completion times of such operations should be determined; constraint sets 16 and 17 schedule the operations that have no precedence relationship.

$$\bar{C}_{ihj'}\geq \bar{C}_{ihj}+\sum_{k'\in M_{ij'}} X_{ihj'k'}\bar{p}_{ij'k'}-A\left(1-Z_{ijj'}\right), \quad \forall i,\ h\in K_i,\quad j,j'\in R_{ih},\ j\neq j' \tag{16}$$

$$\widetilde{C}_{ihj'}\geq \widetilde{C}_{ihj}+\sum_{k'\in M_{ij'}} X_{ihj'k'}\widetilde{p}_{ij'k'}-A\left(1-Z_{ijj'}\right), \quad \forall i,\ h\in K_i,\quad j,j'\in R_{ih},\ j\neq j' \tag{17}$$

For two operations that will be processed by the same machine, the sequence is required to be determined. The basic idea is that if the precedence relationship of any two operations to be processed by the same machine is determined, then all the operations on the same machine will be sequenced

properly without any contradictions. Thus, constraints 18-21 are developed.

$$\begin{aligned}\bar{C}_{i'h'j'} \geq\ & \bar{C}_{ihj} + X_{i'h'j'k'}\bar{p}_{i'j'k'} \\ & - A\left(3 - W_{iji'j'} - X_{ihjk} - X_{i'h'j'k'}\right) \\ & \forall i,\ i',\quad i \neq i',\ h \in K_i,\ h' \in K_{i'},\ j \in R_{ih}, \\ & j' \in R_{i'h'},\ k, k' \in M_{ij} \cap M_{i'j'},\quad k = k' \end{aligned} \tag{18}$$

$$\begin{aligned}\widetilde{C}_{i'h'j'} \geq\ & \widetilde{C}_{ihj} + X_{i'h'j'k'}\widetilde{p}_{i'j'k'} \\ & - A\left(3 - W_{iji'j'} - X_{ihjk} - X_{i'h'j'k'}\right) \\ & \forall i,\ i',\quad i \neq i',\ h \in K_i,\ h' \in K_{i'},\ j \in R_{ih}, \\ & j' \in R_{i'h'},\ k, k' \in M_{ij} \cap M_{i'j'},\quad k = k' \end{aligned} \tag{19}$$

$$\begin{aligned}\bar{C}_{ihj} \geq\ & \bar{C}_{i'h'j'} + X_{ihjk}\bar{p}_{ijk} - A \cdot W_{iji'j'} \\ & - A\left(2 - X_{ihjk} - X_{i'h'j'k'}\right) \\ & \forall i,\ i',\quad i \neq i',\ h \in K_i,\ h' \in K_{i'},\ j \in R_{ih}, \\ & j' \in R_{i'h'},\ k, k' \in M_{ij} \cap M_{i'j'},\quad k = k' \end{aligned} \tag{20}$$

$$\begin{aligned}\widetilde{C}_{ihj} \geq\ & \widetilde{C}_{i'h'j'} + X_{ihjk}\widetilde{p}_{ijk} - A \cdot W_{iji'j'} \\ & - A\left(2 - X_{ihjk} - X_{i'h'j'k'}\right) \\ & \forall i,\ i',\quad i \neq i',\ h \in K_i,\ h' \in K_{i'},\ j \in R_{ih}, \\ & j' \in R_{i'h'},\ k, k' \in M_{ij} \cap M_{i'j'},\quad k = k' \end{aligned} \tag{21}$$

Finally, constraint sets 22-23 determine the makespan.

$$\bar{C}_{\max} \geq \bar{C}_{ihj}, \quad \forall i,\ h \in K_i,\ j \in R_{ih} \tag{22}$$

$$\widetilde{C}_{\max} \geq \widetilde{C}_{ihj}, \quad \forall i,\ h \in K_i,\ j \in R_{ih} \tag{23}$$

Since the same indeterminacy I is adopted in the modeling process, initial test results indicate that the MILP based method is useless due to the NP-hardness of the problem; one cannot obtain even a feasible solution in a reasonable period of time. Especially, compared with MILP model of the deterministic IPPS problem, more variables and constraints are added in the proposed model; this increases the complexity in solving the problem and massive binary variables with complex constraints hinder the application of MILP based method. Therefore, we turn to meta-heuristic algorithms which can capture optimal or near optimal solutions in reasonable time.

IV. VNS BASED MEMETIC ALGORITHM

Moscato proposed the memetic algorithm [72]; it applies local heuristic search method in genetic algorithm to simulate the cultural evolution process. The N5 and Nm neighborhood structures together with VNS local search method are applied in this paper to intensify the search ability. Many successful applications of MA in complex discrete problems optimization have been reported and it is the reason we adopt this swarm intelligence based soft computing technique to address the uncertain IPPS problem. Due to the problem oriented local search method, the proposed MA can achieve more promising scheduling schemes than GA or other evolutionary algorithms.

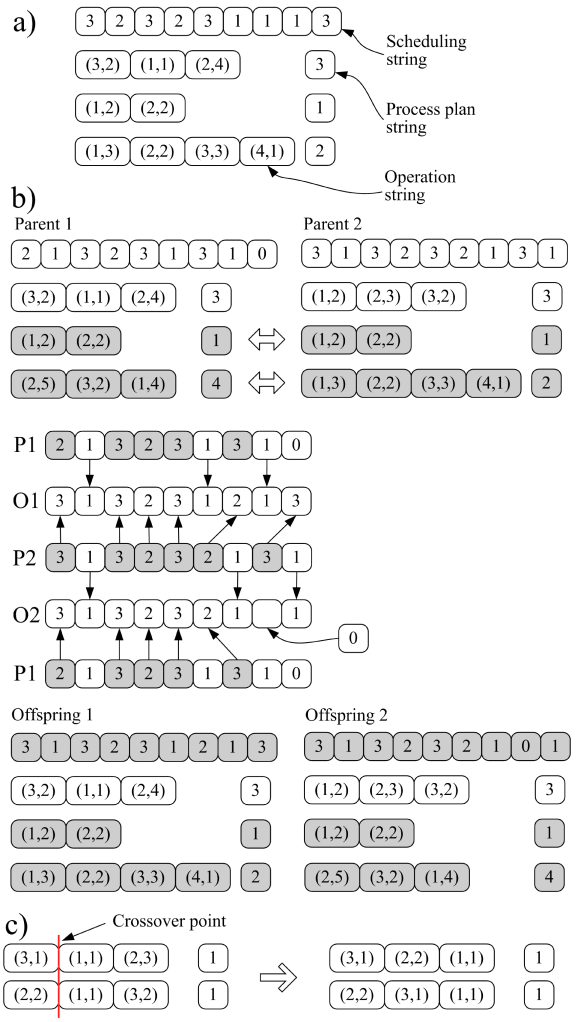


FIGURE 2. Individual and crossover operator.

A. GENETIC OPERATORS

In the proposed MA, each chromosome represents an individual, and it is mapped into a feasible solution through decoding the individual. The coding scheme adopted in this research is a combination of operation combination selection, machine selection, and operation permutation; as depicted in Figure 2a) an individual consists of three parts: the scheduling string, the process plan string, and the operation string.

According to the modelling strategy in Sec. III, there are at least one operation combination, where all the necessary operations to complete a job are available; the only number in the process plan string represents the selected operation combination ID of a job. Each job has an operation string as well as a process plan string; the process plan string is placed at the bottom of each operation string. Operations in a selected operation combination of a job are sequenced properly to follow the precedence requirements indicated in the corresponding network graph by using the binary tree method [73], and the operation ID with the corresponding

machine of an operation are placed in a pair of parentheses in each position in the operation string of that job. The amount of positions in an operation string (the length of an operation string) depends on which operation combination a job will choose and therefore it is equal to $|R_{ih}|$. The number of process plan strings as well as that of operation strings is exactly the amount of jobs to be scheduled in an instance, e.g. $|n|$. The scheduling string in an individual is a permutation of operations IDs; the amount of positions in the scheduling string is $\sum_i |R_{ih}|_{\max}$. For the case when the amount of operations in the selected combination of a job is less than its maximum one, e.g. $|R_{ih}| < |R_{ih}|_{\max}$, corresponding empty positions will be filled with a total of $\sum_i (|R_{ih}|_{\max} - |R_{ih}|)$ 0s.

The proposed coding scheme adopts the operation based paradigm: if number j in scheduling string appears exactly k times, it is the operation that appears in the $k-$th position in the operation string of job j. The advantage of this coding scheme is that it avoids any possible infeasibility during the decoding process. According to the case in Figure 2a), there are three jobs in the instance, and they have 3, 2, and 4 operations respectively with selected operation combination IDs 3, 1, and 2. The number in the third position of the scheduling string is 3; since the number appears for the second time, this means that the operation currently scheduled is in the second position in the operation string of job 3; it is operation 2 of job 3 and it will be processed by machine 2 according to Figure 2a).

The crossover operator includes two parts: the crossover between jobs and the one between process plans of the same job. For two selected parents individuals, randomly select some jobs and exchange the operation strings as well as the process plan strings of the selected jobs; keep other operation strings and process plan strings unchanged. For example, jobs 2 and 3 are selected in Figure 2b) and the operation strings with process plan strings are exchanged. Since different operation combinations of the same job may have distinct numbers of operations, this will affect the scheduling string, and the crossover process (or the patch procedure) on the scheduling string is performed. As illustrated in Figure 2b), the job IDs of the unselected jobs are kept in the same positions; the empty positions of one of the offsprings, e.g. O1 in the figure, are filled with job IDs of the selected jobs in the other parent individual, e.g. P2, as the same sequence in P2; finally, the empty positions are filled with 0s: because job 3 in parent 1 before crossover has three operations, therefore job 3 in offspring O2 also has three operations and a 0 will be filled in the empty position in scheduling string of O2. By this method, two offsprings can be generated. This is the first level crossover operator. The second crossover procedure is performed between two operation stings of the same job with the same combination ID in process plan string; the single point crossover [74], as shown in Figure 2c), is adopted to keep the feasibility of an individual. The operations before the crossover point in both the two strings are kept as they are and the positions after the crossover point are filled with the rest operations that have never appeared before the crossover point in the other string with the corresponding sequence. Such crossover procedure can take the advantage of sequence flexibility, since it allows a set of operations can have various feasible permutations.

In neutrosophic environment, the resultant makespan is also a neutrosophic number; corresponding scheduling method will also be considered. Unfortunately, existing decoding methods are mainly developed for scheduling problems with deterministic processing times. The decoding procedures used in this research are described as follows.

1. Determine the proper processing route of operations according to the scheduling string and the operation string in an individual. The operations will thus be processed one by one.
2. Determine the corresponding machine, e.g. machine k, and neutrosophic processing time, e.g. $\bar{p}_{ijk} + \widetilde{p}_{ijk}I$, for each operation in the processing route.
3. Check every idle time slot on the targeted machine. Suppose the finishing times of the MP of $O_{i'j'}$ and the JP of $O_{ij''}$ are $N_1 = \overline{MT}_k + \widetilde{MT}_k I_1$, and $N_2 = \overline{JT}_{ij'k'} + \widetilde{JT}_{ij'k'} I_2$, respectively; the current operation can be inserted in this time slot only if $\max\{N_1,\ N_2\} + \bar{p}_{ijk} + \widetilde{p}_{ijk}I \leq N_3$, where N_3 is the starting time of the current operation. In particular, if the current operation has no JP or MP, set the starting time of the current operation as $0+0I$ and the finishing time of the operation is $\bar{p}_{ijk} + \widetilde{p}_{ijk}I$.
4. Check if the current operation is the first operation on the corresponding machine; if so, set the starting time of the operation as $0+0I$ and the finishing time of the operation is $\bar{p}_{ijk} + \widetilde{p}_{ijk}I$; otherwise, the starting time can be determined using the maximization operator between the machine available time $\overline{MT}_k + \widetilde{MT}_k I_1$, $I_1 = \left[\beta_1^-, \beta_1^+\right]$ and the job available time $\overline{JT}_{ij'k'} + \widetilde{JT}_{ij'k'} I_2$, $I_2 = \left[\beta_2^-, \beta_2^+\right]$ (operation $O_{ij'}$ is processed before O_{ij} and machine k' is usually not exactly the machine k): the starting time of the operation O_{ij} is $\overline{ST}_{ijk} + \widetilde{ST}_{ijk}I = [\max\{\overline{MT}_k + \widetilde{MT}_k\beta_1^-, \overline{JT}_{ij'k'} + \widetilde{JT}_{ij'k'}\beta_2^-\}, \max\{\overline{MT}_k + \widetilde{MT}_k\beta_1^+, \overline{JT}_{ij'k'} + \widetilde{JT}_{ij'k'}\beta_2^+\}]$; based on the starting time, the completion time of the operation O_{ij} can be calculated by using the addition operator of neutrosophic numbers.
5. Return to Step 2. and schedule operations one by one till all the operations are assigned to machines properly.

An illustrative example is given here for a better understanding of the active scheduling based decoding procedure. There are three machines and three jobs in Figure 3. The starting time of the first operation on each machine is $0+0I$, and therefore operations $J1.1$, $J2.1$, $J3.1$ can be processed on time 0; however, their finishing times are neutrosophic numbers and the fluctuations are marked in the figure. In assigning operation $J1.3$ on machine $M3$, this operation should be started after both its machine predecessor $J3.1$ and job predecessor $J1.2$; according to the maximization operator of neutrosophic numbers, the starting time of operation $J1.3$

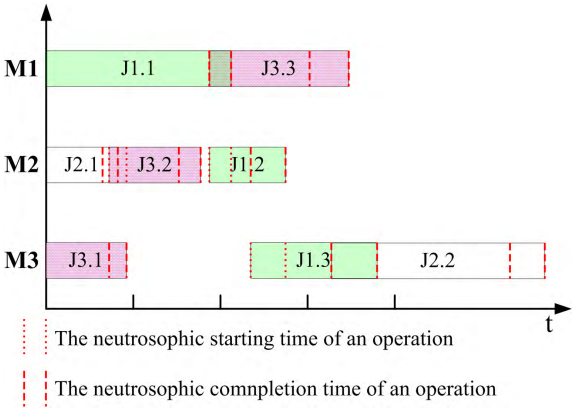


FIGURE 3. An example of decoding.

is determined by the finishing time of operation $J1.2$ because the finishing time of operation $J1.2$ is later than that of operation $J3.1$. In addition, operation $J3.2$ is assigned on machine $M2$ and the time slot between operations $J2.1$ and $J1.2$ is checked to judge whether it is appropriate to insert operation $J3.2$ in this slot. After calculating the possible finishing time of operation $J3.2$ and applying the ranking method, this time slot is feasible and operation $J3.2$ is thus inserted between operations $J2.1$ and $J1.2$; otherwise, it can only be appended after operation $J1.2$. Operation $J2.2$ is not so lucky to be inserted in the time slot between operations $J3.1$ and $J1.3$ since the processing time of this operation is quite long and the time slot between operations $J3.1$ and $J1.3$ cannot hold it.

Such active scheduling method where operations may properly be inserted in feasible time slots can effectively shorten the makespan, since makespan is also an important criterion in this research: a scheduling scheme that considers only the robustness criterion may increase the completion times of jobs. However, existing researches reveal that only the active scheduling method is not enough to shorten the makespan. In the following section, the N5 and Nm neighborhoods based VNS local search method will be introduced to intensify the search ability.

The two criteria, e.g. the nominal makespan as well as the robustness of a schedule (the deviation of makespan values), are considered in a weighted sum manner since they are both quite crucial in real life production situations. Since the makespan is a neutrosophic number and can be obtained by the ranking method, the deviation value is thus exactly the fluctuation range of makespan and it equals $(\bar{C}_{\max} + \widetilde{C}_{\max}\beta^{+}) - (\bar{C}_{\max} + \widetilde{C}_{\max}\beta^{-}) = \widetilde{C}_{\max}(\beta^{+} - \beta^{-})$, where $I = [\beta^{-}, \beta^{+}]$. Apparently, a solution is more robust provided that it has a smaller value of deviation. The selection method should consider both of the two criteria simultaneously: we present here a normalized weighted sum based tournament selection method to distinguish a more promising individual. In this method, values of the two criteria of each individual are divided by the global worst ones respectively; the total fitness of an individual can be calculated as $w\times$criterion1$+(1-w)\times$criterion2. Therefore, an individual with a smaller sum after addition of the two normalized values will be more promising.

B. THE VNS LOCAL SEARCH METHOD

The nominal makespan is also quite important and a robust scheduling scheme with large makespan value is unacceptable because this may delay the subsequent production plan and reduce the machine utilization. The local search methods are thus considered to shorten the nominal makespan. As an important part of MA, the local search technique is quite important in improving the solution quality because the solution quality improvement by meta-heuristic algorithm is limited. Unfortunately, existing researches on scheduling problems with uncertain processing times usually consider no local search methods in optimization algorithms or just applying rough and aimless local search methods. Therefore, more sophisticated and problem oriented local search method is required for further exploitation search for a single individual. On the other hand, many existing local search methods have usually a few parameters and the optimization effect mainly relies on the parameters settings; this hinders wide applications of such local search methods. This research adopts the VNS local search technique to perform the local search procedure. The VNS algorithm is a simple while effective and powerful local search methodology with less parameters. By systematic jumps from one neighborhood structure to the other, the VNS local search procedure can effectively help an individual to avoid being trapped into local optimum through performing some perturbations (or moves) on an individual; this is totally different from the traditional local search procedure where the only optimization trajectory on an individual is followed till there is no further improvement. The outstanding feature of the VNS search procedure is that it is capable to change the current search direction or search trajectory in search space and a better solution may be obtained from a distinct perspective.

It has been observed that the makespan value of a scheduling scheme is determined by the critical path [32], which contains the operations each can affect the total completion time of an schedule. Therefore, to shorten the makespan, one should reconstructing the critical path by reordering the critical operations in the critical block or shifting the critical operations to other machines to break existing critical path; otherwise, the makespan cannot be shortened. In N5 neighborhood structure, swaps of two neighbouring operations near the borderline of a critical block is performed. A critical path u of a schedule $G(\pi)$ is composed of some segments of (partial) operation permutations on the machines, e.g. $\pi_k, k = 1, 2, \ldots K$, and segments of such (partial) operation permutations are called critical blocks $B_1, B_2, \ldots, B_r$. Nowicki and Smutnicki have shown that the size of the N5 neighborhood is reduced by abandoning the moves that will not directly improve the makespan; therefore, only the moves where swapping the first two or (and) last two operations in a block is considered:

- For the first block B_1, only the last two operations are swapped provided that B_1 contains at least two critical operations;
- For the last block B_r, only the first two operations are swapped provided that B_r contains at least two critical operations;
- For other blocks that contains at least two critical operations, the first two and the last two operations can be swapped.

For the schedule scheme in the Gantt chart in Figure 4 a), the critical path contains three blocks: $B_1 = \{O3.1\}$, $B_2 = \{O3.2, O2.1\}$, $B_3 = \{O2.2, O3.3, O1.4\}$; since B_1 contains only one operation, the moves of swapping two operations can be considered for other two critical blocks and the possible moves are: 1) swapping operations $O3.2$, $O2.1$ in B_2 and 2) swapping operations $O2.2$, $O3.3$ in B_3. Figure 4 b) is the corresponding Gantt chart where the move of swapping operations $O3.2$, $O2.1$ is performed; clearly, the nominal makespan has been shortened.

The Nm neighborhood structure takes the advantage of the operation flexibility: it removes a critical operation to another available machine to shorten the critical path u. A critical operation v in u can be rescheduled on another machine k, if it follows the correct precedence relationships: v can be reinserted after its job predecessor and before its job successor. In Figure 4 a), the makespan is solely determined by the critical path ($O3.1 - O3.2 - O2.1 - O2.2 - O3.3 - O1.4$) (shaded operations) and noncritical operations can be started at the latest starting times without increasing the makespan. Operations $O2.1$ and $O2.2$ in Figure 4 c) are reinserted in machines M2 and M3 respectively; note that noncritical operation $O1.3$ in Figure 4 c) can be started later as long as it is started before the starting time of $O1.4$. The new scheduling scheme is illustrated in Figure 4 d), where the new makespan value is reduced.

Instead of aimless local search methods in other publications, the VNS search method is problem oriented and the later experimental results also reflects its effectiveness. Algorithm 2 presents the work flow of the VNS search procedure. The search procedure begins from the last critical block till it encounters the first one. If a solution cannot be improved by a neighborhood structure, it jumps to the other one until there is no further improvement. Whenever the individual is improved, a new critical path is determined and search procedure continues.

C. THE WORK FLOW OF MA

The main work flow is given in Figure 5. The genetic operators are performed at first in each iteration, and the VNS local search procedure is performed later for each individual. The iteration continues till the stop criterion is meet. Since both the nominal makespan as well as the robustness criteria are considered, the global best solution updating method, which strikes a balance between the two criteria, is developed as follows.

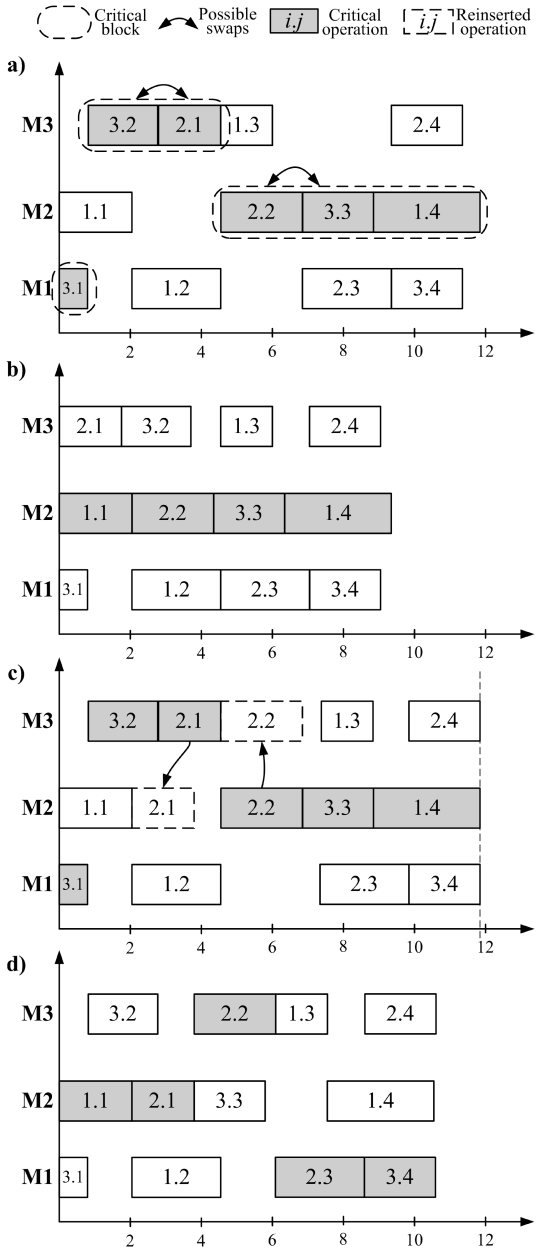


FIGURE 4. Examples of the neighborhood structures: a) the Gantt chart of the original schedule; b) the Gantt chart of the optimized schedule using the N5 neighborhood structure; c) Determination of the possible moves; and d) the optimized scheduling scheme using the Nm neighborhood structure.

- Identify the maximum nominal makespan $C_{1\max}$ and the maximum deviation on completion times as $C_{2\max}$ in both the global best individual *GBest* and current best individual *CBest*.
- Calculate the normalized fitness values using equation $Fit = w(nominalmakespan/C_{1\max}) + (1 - w)(deviation/C_{2\max})$ for both *GBest* and *CBest*.
- If $Fit_{GBest} > Fit_{CBest}$ replace *GBest* with *CBest*.

V. EXPERIMENTAL STUDY

The well-known Kim's benchmark instances are adopted in the experiments; this set of 24 instances covers small,

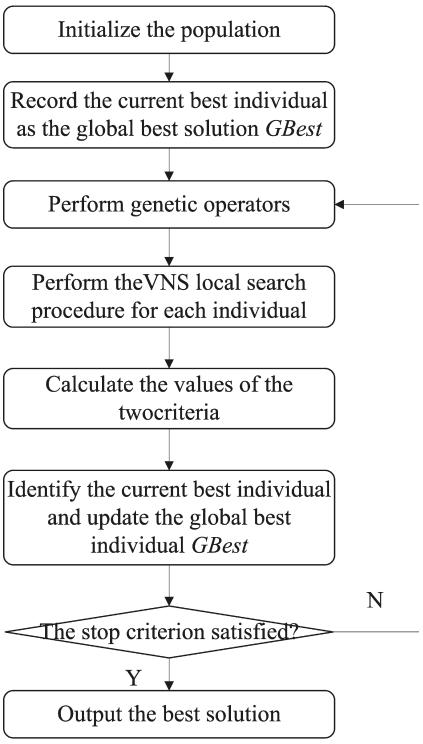


FIGURE 5. The work flow of the proposed algorithm.

medium and large scale instances and they are to be processed on 15 different machines. As mentioned above, the robustness criterion in the algorithm is defined as the maximum variation on makespan. In order to simulate different situations of the uncertainty in processing times, three situations of fluctuations in processing times, e.g. small fluctuations, medium fluctuations, and large fluctuations, are considered by setting the factor α to 0.05, 0.15 and 0.5 respectively; the α values are used to reflect different levels of uncertainties in processing times and a certain α value gives the upper and low bounds of the uncertain processing time: $\mathcal{R} = [p_{ijk} - \alpha p_{ijk},\ p_{ijk} + \alpha p_{ijk}]$, where p_{ijk} is the nominal processing time of O_{ij}. Since processing times of different operations will actually fall within distinct ranges $\mathcal{R}'$, $\mathcal{R}' \subseteq \mathcal{R}$, for better simulations of processing times of operations, the range that the actual processing time of operation O_{ij} on machine k will fall within can be set as $\mathcal{R}' = [p_{ijk} - u_{ijk}\alpha p_{ijk},\ p_{ijk} + u_{ijk}\alpha p_{ijk}]$, where u_{ijk} is a coefficient by uniformly sampling in the interval $[0, 1]$ to place the actual restrictions on low and upper bounds of processing times. Therefore, it follows that $\bar{p}_{ijk} + \tilde{p}_{ijk}I = \mathcal{R}'$ and $\bar{p}_{ijk'} + \tilde{p}_{ijk'}I = \mathcal{R}''$. For example, suppose $\alpha = 0.5$ and the nominal processing times of an operation on machines k and k' are 20 and 10 respectively, the actual process times will fall in ranges $[15,\ 25]$ and $[5,\ 15]$ with $u_{ijk} = 0.5$ and $u_{ijk'} = 1$ respectively. Further, the range $\mathcal{R}'$ can be easily transformed into a neutrosophic number and thus uncertain processing times used in the algorithm are available.

The proposed algorithm is coded in C++ language and performed on a computer equipped with an Intel i5-9600 3.7GHz CPU and 16GB memory. 800 individuals are employed and the algorithm will be stopped after 1000 iterations. Based on the initial trials, the crossover probability is

Algorithm 2 The VNS Search Procedure

Require: 1)Two neighborhood structures: N5 and Nm; 2)Initial solution S_0 with makespan $\bar{C}_{\max}$.
Ensure: The optimal solution S_{opt}
while 1 **do**
  $IsStop1 := false$
  while !$IsStop1$ **do**
    Set $StopN5 := false$. Set $EndPath := false$. Determine a critical path and critical blocks.
    while !$StopN5$ **do**
      If block B_i is the last block and has more than one operations, swap just the first two operations; if B_i is the first block, set $EndPath := true$ and swap just the last two operations if the block has more than one operations. For other cases, if B_i has more than one operations, swap the last two operations first and then the first two operations.
      Calculate the makespan after swapping: $\bar{C}'_{\max}$.
      if $\bar{C}'_{\max} < \bar{C}_{\max}$ **then**
        Accept the move; update S with the current solution and set $StopN5 := true$, $IsStop1 := false$.
      end if
      if $StopN5 == false$ and $EndPath == true$ **then**
        Set $IsStop1 := true$ and $StopN5 := true$.
      end if
    end while
  end while Record current solution as S_1; set $IsStop2 := false$.
  while !$IsStop2$ **do**
    Obtain the critical path and critical operations. Set the initial makespan as $\bar{C}_{\max}$. Set $StopNm := false$ and $EndPath := false$.
    while !$StopNm$ **do**
      Insert each critical operation into a proper position on an alternative machine and calculate the makespan $\bar{C}'_{\max}$. If all the critical have been processed, set $EndPath := true$.
      if $\bar{C}'_{\max} < \bar{C}_{\max}$ **then**
        Accept the move; update S_1 with the current solution S_2 and set $StopNm := true$, $IsStop2 := false$.
      end if
      if $StopNm == false$ and $EndPath == true$ **then**
        Set $StopNm := true$ and $IsStop2 := true$.
      end if
    end while
  end while
  if S_1 is better than current solution S_2 **then**
    Set $S_1 := S_{opt}$ and **break** the while loop to stop the VNS procedure.
  end if
end while

set as 0.7. In each computation, coefficients u_{ijk}s are generated at first and 10 independent computations are performed for each instance; the nominal makespan as well as the robustness criterion will be analyzed based on the results of 10 computations.

In order to reflect the preference on the two criteria, the weights are set to 0.25, 0.5, and 0.75 respectively; according to Sec. IV-C, a small weight value means the algorithm puts more emphasis on the robustness criterion and the algorithm treats the two criteria equally with a weight value 0.5. In the following, we first give an intuitive presentation on the convergence as well as the solution quality of the most complicated instance (Instance 24) with weight $w = 0.5$. As depicted in Figure 6 a), the VNS local search procedure presents the powerful search ability in a small fluctuation factor case ($\alpha = 0.05$): the nominal makespan is about 500 time units while this value is more than 550 if the VNS search is not considered; therefore, the VNS search procedure is quite necessary in such cases because a robust scheduling scheme with a deteriorating makespan value cannot deal with varied challenges in modern manufacturing and production. Figure 6 b) gives the convergence curves with a large α value; it means that processing time fluctuations become larger. The nominal makespan value obtained by the proposed VNS based MA is still a little better than the case where the VNS local search method is not considered. It is noteworthy that a compact operation permutation in a scheduling scheme (or Gantt chart) indicates there is less or no idle time between operations on a machine to absorb the actual processing time variations and consequently, a small makespan value is more sensitive to processing time variations (or processing time uncertainties); hence, a large time deviation on makespan deteriorates the robustness criterion. On the contrary, a scheduling scheme with a large nominal makespan value may perform better in robustness criterion. This reflects the conflict of the two criteria. In Figure 6 b), the robustness of makespan obtained by the VNS based MA is worse than the one obtained by the plain genetic algorithm.

As discussed in Sec. IV-C, since the global best solution updating method considers both the two criteria simultaneously, the newly updated solution will certainly be better than the old one in general although one of its criterion, e.g. the nominal makespan criterion, may be a little worse than the old one; therefore, values of the curves become larger than the one in some iterations before. In addition, we can also see from Figure 6 that the fluctuations of makespan values in both the two cases become smaller as the optimization process goes on, and this reflects the effectiveness of the proposed algorithm: it can really make solutions become more robust.

As analyzed in Sec. IV-B, the determinstic makespan $C_{\max}$ depends on the length of the critical path. Suppose the critical path is not changed in uncertain processing time environment, the lower and the upper bounds of the makespan are $(1-\alpha)C_{\max}$ and $(1+\alpha)C_{\max}$ respectively; in other words, the fluctuation of the actual makespan will also be $\pm\alpha \times \%$. To compare or estimate a robust scheduling scheme, two metrics,

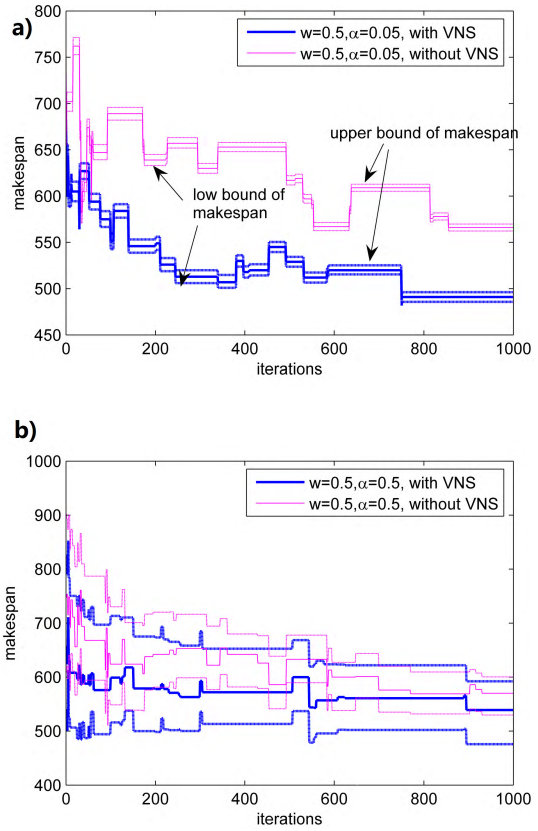


FIGURE 6. Convergence curves of Instance 24.

namely the upper deviation (UD) and lower deviation (LD), are defined as:

$$LD = \frac{|nominal\ makespan - low\ bound\ of\ makespan|}{nominal\ makespan} \tag{24}$$

$$UD = \frac{|nominal\ makespan + upper\ bound\ of\ makespan|}{nominal\ makespan} \tag{25}$$

A solution with smaller values of UD and LU is much more robust and a solution with a smaller value of nominal makespan (NM) means the scheduling scheme is more efficient. Computational results of Kim's benchmark instances with weight $w = 0.5$ under fluctuation factor $\alpha = 0.05$ are listed in Table 1.

The performance of the proposed MA is compared with the plain GA (MA without VNS) in the table. For the mean values of NM, LD, and UD in both the two cases, the better ones are shown in bold. The merit of applying VNS based local search procedure is reflected: mean values of nominal makespan obtained by MA is much better that those obtained by plain GA: MA performs better than GA for 14 instances while GA performs better than MA for only 8 instances from Table 1. More importantly, it can be perceived from Table 1 that large scale instances, e.g. Instances 22, 23 and 24, receive more improvements in the nominal makespan

TABLE 1. Results comparisons between the two algorithms with $w = 0.5$, $\alpha = 0.05(5\%)$.

	The proposed MA			MA without VNS		
Cases	NM[a](B,W,M)[b]	LD(B,W,M)(%)	UD(B,W,M)(%)	NM(B,W,M)	LD(B,W,M)(%)	UD(B,W,M)(%)
1	(446,495,462.4)	(0.44,0.78,**0.59**)	(0.44,0.79,**0.59**)	(432,483,**452.9**)	(0.42,1.02,0.76)	(0.42,1.02,0.76)
2	(347,408,**370.6**)	(0.36,1.00,0.66)	(0.37,1.05,0.66)	(358,408,376.9)	(0.41,0.91,**0.64**)	(0.41,0.91,**0.64**)
3	(354,393,**371.7**)	(0.44,0.71,**0.58**)	(0.45,3.07,0.83)	(349,399,376.3)	(0.43,1.01,0.71)	(0.43,1.01,**0.71**)
4	(310,330,**321.4**)	(0.53,1.11,0.72)	(0.46,0.98,0.66)	(312,347,323.3)	(0.47,0.77,**0.61**)	(0.47,0.77,**0.61**)
5	(330,371,342.3)	(0.46,0.79,**0.58**)	(0.46,0.79,**0.59**)	(325,353,**341.1**)	(0.46,0.89,0.67)	(0.47,0.89,0.68)
6	(447,470,460.9)	(0.31,0.93,**0.62**)	(0.31,0.93,**0.58**)	(430,473,**457.3**)	(0.50,0.84,0.63)	(0.50,0.84,0.63)
7	(372,406,387.6)	(0.52,1.00,**0.80**)	(0.52,1.00,0.80)	(375,402,**387.0**)	(0.46,1.49,0.89)	(0.42,1.25,**0.74**)
8	(355,409,374.8)	(0.45,0.75,**0.56**)	(0.45,0.75,**0.56**)	(346,410,374.8)	(0.35,0.88,0.71)	(0.35,0.88,0.71)
9	(442,485,462.3)	(0.44,0.69,0.57)	(0.44,0.69,0.57)	(430,472,**452.4**)	(0.29,0.76,0.57)	(0.29,0.76,0.57)
10	(430,520,464.7)	(0.38,0.74,**0.58**)	(0.38,0.74,**0.59**)	(439,479,**455.1**)	(0.48,1.05,0.67)	(0.48,0.78,0.65)
11	(363,429,**389.3**)	(0.39,0.71,**0.57**)	(0.40,0.81,**0.59**)	(375,428,391.7)	(0.45,0.92,0.66)	(0.46,0.92,0.67)
12	(322,366,**341.1**)	(0.38,1.12,0.70)	(0.12,0.82,**0.54**)	(332,399,356.7)	(0.43,0.84,**0.67**)	(0.44,0.84,0.67)
13	(451,486,464.8)	(0.44,1.04,**0.64**)	(0.44,0.90,**0.60**)	(440,488,**461.3**)	(0.44,1.05,0.70)	(0.44,0.78,0.67)
14	(379,425,**395.1**)	(0.45,1.13,0.79)	(0.05,0.93,**0.55**)	(401,438,418.9)	(0.53,1.01,**0.75**)	(0.53,1.26,0.81)
15	(439,473,458.2)	(0.33,1.12,**0.58**)	(0.22,0.64,**0.51**)	(429,470,**451.0**)	(0.32,0.91,0.68)	(0.32,0.91,0.68)
16	(452,492,**464.3**)	(0.40,1.18,**0.68**)	(0.12,0.79,**0.53**)	(447,492,466.3)	(0.47,0.96,0.70)	(0.43,0.96,0.66)
17	(387,438,**406.9**)	(0.34,0.80,**0.59**)	(0.34,0.95,**0.62**)	(401,452,425.8)	(0.46,0.93,0.70)	(0.46,0.93,0.70)
18	(356,390,**375.0**)	(0.35,0.65,**0.49**)	(0.14,0.57,**0.43**)	(358,424,393.0)	(0.48,0.95,0.67)	(0.48,0.95,0.68)
19	(447,513,**475.5**)	(0.41,1.44,**0.66**)	(0.41,0.72,**0.58**)	(458,499,476.6)	(0.31,0.93,0.69)	(0.31,0.93,0.69)
20	(401,463,**420.4**)	(0.36,0.91,0.71)	(0.36,0.84,0.67)	(423,501,460.6)	(0.50,0.83,**0.65**)	(0.50,0.83,**0.65**)
21	(446,486,**464.7**)	(0.40,0.80,**0.55**)	(0.40,0.80,**0.55**)	(449,524,479.3)	(0.38,1.73,0.56)	(0.38,0.73,0.56)
22	(462,516,**493.7**)	(0.37,0.94,0.72)	(0.12,0.85,**0.56**)	(483,547,515.8)	(0.50,0.87,**0.65**)	(0.50,0.92,0.66)
23	(439,486,**453.0**)	(0.40,0.73,**0.62**)	(0.29,0.73,**0.59**)	(458,523,492.3)	(0.51,1.11,0.76)	(0.25,0.94,0.61)
24	(491,532,**514.6**)	(0.41,1.14,0.80)	(0.39,1.05,0.72)	(514,628,575.2)	(0.55,0.96,**0.73**)	(0.55,0.88,**0.69**)

[a] NM: nominal makespan.
[b] B, W, M: The best, the worst, and the mean values respectively.

criterion after applying the VNS based memetic algorithm; the reason behind this is that there are more operations in large scale instances and conventional meta-heuristics, e.g. plain GA, is incapable to perform individual based subtle search procedures. Without VNS based local search method in this research, the plain GA cannot yield solutions with competitive nominal makespan values. As discussed in Sec IV-C, the global best solution updating method considered both the two criteria simultaneously and equally, and the algorithm usually abandons solutions with competitive nominal makespan values only but chooses a solution which can achieve a balance between the two criteria as the best solution. For some small scale instances, e.g. Instances 5, 6 and 7, it seems that the plain GA yields competitive solutions for the nominal makespan criterion according to Table 1; but indeed deviation values (LD and UD values) of MA are better than those of plain GA for these instances.

According to Table 1, mean UD and LD values are all less than the fluctuation factor α; this reveals that the solutions obtained by MA and GA are quite robust. Obviously, the so-called optimal solutions obtained by novel algorithms for the deterministic IPPS problem may performs poorly with uncertain processing times. Therefore, the optimization method that once applied to solve the deterministic IPPS problem cannot be directly applied to solve the uncertain IPPS problem. Based on the observation on LD and UD values of both MA and GA (MA without VNS) in Table 1, deviations of actual makespan of MA is much better than those of GA, and therefore the resultant solutions of MA are more robust and they can absorb certain processing time fluctuations. In the following, we analyze the solutions obtained by both MA and GA with large processing time fluctuations (with weight $w = 0.5$ under fluctuation factor $\alpha = 0.5(50\%)$) and corresponding results are summarized in Table 2.

Similar situations can be observed in Table 2: nominal makespan values obtained by MA are generally better than those yielded by the plain GA and this also reveals that the proposed VNS is powerful in shortening makespan values for complicated scheduling instances despite large or small fluctuations in processing times. For the robustness criterion, as presented by the LD and UD values, resultant solutions of MA have smaller LD and UD values; therefore, the maximum completion times of these scheduling schemes have smaller fluctuation ranges. Table 2 also shows the talent and strength of the proposed memetic algorithm: the maximum LD and UD values are all less than 10% while the fluctuation factor α of processing times is 50%. In other words, the fluctuation of maximum completion time is smaller than the fluctuation of the processing time of a single operation. Meanwhile, computational results in both the two tables also indicate that considering only the (nominal) makespan criterion is usually not enough to cope with common processing time fluctuations in real life production situations.

We analyze here the cases with different weights; besides the above cases with $w = 0.5$, two weights, $w = 0.25$ and $w = 0.75$, are assigned and corresponding results will be studied. The algorithm puts more emphasis on the nominal makespan criterion provided that a larger weight is assigned and conversely, a smaller weight means the proposed algorithm lays particular stress on the robustness of the resultant

TABLE 2. Results comparisons between the two algorithms with $w = 0.5$, $\alpha = 0.5(50\%)$.

	The proposed MA			MA without VNS		
Cases	NM[a](B,W,M)[b]	LD(B,W,M)(%)	UD(B,W,M)(%)	NM(B,W,M)	LD(B,W,M)(%)	UD(B,W,M)(%)
1	(427,495,**462.6**)	(6.60,12.34,**8.33**)	(0.83,9.11,**6.93**)	(429,506,472.0)	(5.47,11.72,8.39)	(4.38,11.73,7.55)
2	(355,424,383.3)	(5.98,12.12,8.35)	(0.47,9.13,6.48)	(346,401,**366.7**)	(5.69,11.46,**7.84**)	(1.86,9.46,**5.54**)
3	(355,388,**371.6**)	(5.95,11.86,**9.03**)	(1.50,9.61,**5.47**)	(357,431,378.3)	(6.26,14.22,9.39)	(1.16,10.25,6.18)
4	(307,343,**322.9**)	(6.27,11.26,**8.56**)	(0.42,9.34,6.47)	(306,355,333.9)	(6.64,14.31,9.84)	(1.89,8.23,**5.04**)
5	(329,383,354.2)	(4.17,10.61,**7.52**)	(1.44,9.35,5.62)	(320,389,**351.5**)	(4.11,13.95,8.32)	(2.58,8.71,**5.61**)
6	(443,507,**476.5**)	(5.48,9.62,**7.08**)	(1.78,8.51,**5.59**)	(455,504,479.6)	(4.83,10.65,7.46)	(4.20,10.39,7.22)
7	(374,414,392.5)	(5.66,12.48,**9.15**)	(5.73,13.03,8.90)	(377,420,**389.4**)	(6.87,15.29,12.15)	(3.74,12.53,**7.66**)
8	(357,424,**378.1**)	(5.66,12.53,**7.59**)	(4.11,8.96,**5.94**)	(350,410,379.3)	(5.67,11.81,7.97)	(2.67,10.43,6.83)
9	(451,510,480.9)	(6.31,10.63,8.08)	(0.43,8.39,**4.98**)	(430,501,**472.8**)	(4.49,13.67,**7.38**)	(3.20,8.01,6.09)
10	(446,521,484.4)	(5.26,7.76,**6.19**)	(4.65,7.36,**6.12**)	(429,515,**471.1**)	(5.49,11.14,7.92)	(3.53,8.59,6.32)
11	(363,405,**390.9**)	(4.09,9.23,**7.18**)	(1.79,10.48,**5.81**)	(390,430,403.2)	(5.79,12.35,8.88)	(1.38,12.86,6.41)
12	(330,393,359.9)	(4.93,12.05,**8.02**)	(0.51,10.88,5.51)	(330,391,**356.8**)	(5.19,11.69,8.19)	(1.63,8.47,**5.27**)
13	(451,497,**475.5**)	(4.61,8.26,**6.82**)	(4.61,8.49,**6.98**)	(464,520,486.2)	(5.15,9.57,7.18)	(3.65,8.96,7.14)
14	(392,430,**415.2**)	(6.69,11.08,8.20)	(5.00,11.45,8.09)	(402,463,427.4)	(5.57,10.14,**7.57**)	(2.61,8.47,**6.73**)
15	(446,496,477.4)	(4.58,8.71,**6.20**)	(2.90,7.42,**5.57**)	(437,510,**477.3**)	(4.40,8.93,7.09)	(4.40,9.34,6.73)
16	(446,492,**472.9**)	(4.85,9.00,**7.17**)	(5.81,8.64,**7.24**)	(466,528,496.0)	(6.67,10.24,8.00)	(5.87,8.86,7.27)
17	(408,464,**424.4**)	(4.53,9.46,**7.28**)	(2.65,10.35,7.08)	(420,481,447.8)	(5.80,9.41,7.37)	(5.24,9.91,7.08)
18	(369,406,**380.1**)	(5.27,10.07,7.26)	(1.31,8.25,**5.25**)	(375,438,404.7)	(4.78,10.81,**6.96**)	(2.94,7.87,5.76)
19	(451,517,**496.0**)	(4.78,9.74,7.08)	(3.74,10.01,**7.03**)	(463,533,510.5)	(4.73,8.52,**6.77**)	(5.90,10.19,7.43)
20	(430,468,**445.1**)	(5.30,8.92,**6.92**)	(3.79,8.99,**6.83**)	(425,506,459.5)	(5.92,11.42,8.32)	(3.71,10.59,7.58)
21	(447,510,**485.1**)	(4.74,8.78,**6.76**)	(2.14,7.98,**6.02**)	(459,512,493.2)	(4.85,8.40,6.86)	(4.86,8.40,6.84)
22	(466,538,**511.2**)	(5.78,8.65,7.03)	(5.57,9.31,7.28)	(520,584,549.8)	(4.36,9.81,**6.76**)	(5.78,9.81,**7.25**)
23	(449,508,**472.9**)	(5.98,9.93,**7.87**)	(5.51,9.94,8.29)	(475,550,505.7)	(5.43,11.54,8.01)	(4.46,10.91,**7.60**)
24	(534,579,**550.9**)	(5.24,11.72,**7.36**)	(5.58,9.87,7.91)	(541,613,585.6)	(6.12,8.42,7.52)	(5.32,8.47,**7.24**)

[a] NM: nominal makespan.
[b] B, W, M: The best, the worst, and the mean values respectively.

TABLE 3. Results comparisons with distinct weights under small processing time fluctuations ($\alpha = 0.05(5\%)$).

	$w = 0.25$			$w = 0.5$			$w = 0.75$		
Cases	NM[a](M[b])	LD(M)(%)	UD(M)(%)	NM(M)	LD(M)(%)	UD(M)(%)	NM(M)	LD(M)(%)	UD(M)(%)
1	482.1	0.59	0.59	462.4	0.59	0.59	441.1	0.91	0.88
2	379.6	0.54	0.54	370.6	0.66	0.66	354.9	0.72	0.71
12	351.5	0.62	0.51	341.1	0.70	0.54	331.9	0.81	0.63
13	477.6	0.60	0.61	464.8	0.64	0.60	441.4	0.85	0.75
23	471.1	0.56	0.56	453.0	0.62	0.59	429.8	0.84	0.79
24	558.1	0.65	0.62	514.6	0.80	0.72	482.5	0.95	0.90

[a] NM: nominal makespan.
[b] M: The mean values.

solution. For the sake of simplicity, partial instances that stand for small, medium and large scale instances are chosen in the following study: Instances 1-2, 12-13, and 23-24 are selected to represent the small, medium and large scale instances. Table 3 gives the results of these instances with distinct weights ($w = 0.25, 0.5, 0.75$) under small fluctuation factor ($\alpha = 0.05(5\%)$). Clearly, Table 3 reveals that the weight factor is quite important in solution updating procedure and the selection operator: by rationally adjusting the weight factor, one can compromise between the two criteria and obtain solutions according to one's preference. By comparing with data in columns 2-4 and columns 8-10 in Table 3, it shows that mean values of the nominal makespan of weight 0.75 is much better than those of 0.25; on the contrary, the mean values of LD and UD of weight 0.25 are generally less than those of weight 0.75. Therefore, to reach a compromise between the two criteria, the weight can be set as $w = 0.5$ as shown in columns 5-7 in Table 3. For cases of medium and large processing time fluctuations, corresponding data are summarized in Tables 4 and 5. As the situation in Table 3, the weight factor regulates the preference of the two criteria. Clearly, the algorithm can thrashed out a compromise with a wight of 0.5. For a smaller value of the weight, the yielded solutions have better robustness; for instance, the LD and the UD values of Instance 24 with the weight factor $w = 0.25$ in Tables 4 and 5 are smaller than the ones of the same instance with the weight factor $w = 0.75$. For the nominal makespan criterion, similar with the situations in Table 3, its performance also depends on the weight factor. In Table 5, the average value of nominal makespan of Instance 24 is only 510.6 with weight 0.75; however, the value increases to 583.7 when the weight is set to 0.25. From the analysis above with the data in Tables 3, 4 and 5 we can see that the nominal makespan criterion and the robustness criterion are two conflict objectives.

Different with single objective case, the proposed MA should strike a balance between the two criteria, and hence the algorithm falls into a time consuming seesaw battle.

TABLE 4. Results comparisons with distinct weights under medium processing time fluctuations ($\alpha = 0.15(15\%)$).

	$w = 0.25$			$w = 0.5$			$w = 0.75$		
Cases	NM[a](M[b])	LD(M)(%)	UD(M)(%)	NM(M)	LD(M)(%)	UD(M)(%)	NM(M)	LD(M)(%)	UD(M)(%)
1	495.4	1.99	2.01	465.3	2.47	1.91	452.0	2.44	2.29
2	376.4	2.02	1.67	370.8	1.83	1.92	357.3	2.69	2.28
12	362.8	1.95	1.66	349.0	1.93	1.94	334.7	2.68	2.28
13	488.3	1.99	1.83	465.5	2.18	1.82	448.1	2.89	1.99
23	497.3	1.92	1.94	450.6	2.04	2.13	426.9	2.67	2.56
24	570.6	1.86	1.77	524.7	2.07	2.05	491.6	2.85	2.39

[a]NM: nominal makespan.
[b]M: The mean values.

TABLE 5. Results comparisons with distinct weights under large processing time fluctuations ($\alpha = 0.5(50\%)$).

	$w = 0.25$			$w = 0.5$			$w = 0.75$		
Cases	NM[a](M[b])	LD(M)(%)	UD(M)(%)	NM(M)	LD(M)(%)	UD(M)(%)	NM(M)	LD(M)(%)	UD(M)(%)
1	482.9	8.07	6.05	462.6	8.33	6.93	448.6	9.98	7.18
2	379.7	6.73	5.18	383.3	8.35	6.48	359.4	9.38	6.31
12	363.5	7.33	5.17	359.9	8.02	5.51	343.0	8.76	6.89
13	498.2	6.41	5.03	475.5	6.82	6.98	462.5	7.70	7.56
23	501.9	6.87	6.76	472.9	7.87	8.29	446.0	7.72	8.40
24	583.7	5.87	6.21	550.9	7.36	7.91	510.6	8.50	9.50

[a]NM: nominal makespan.
[b]M: The mean values.

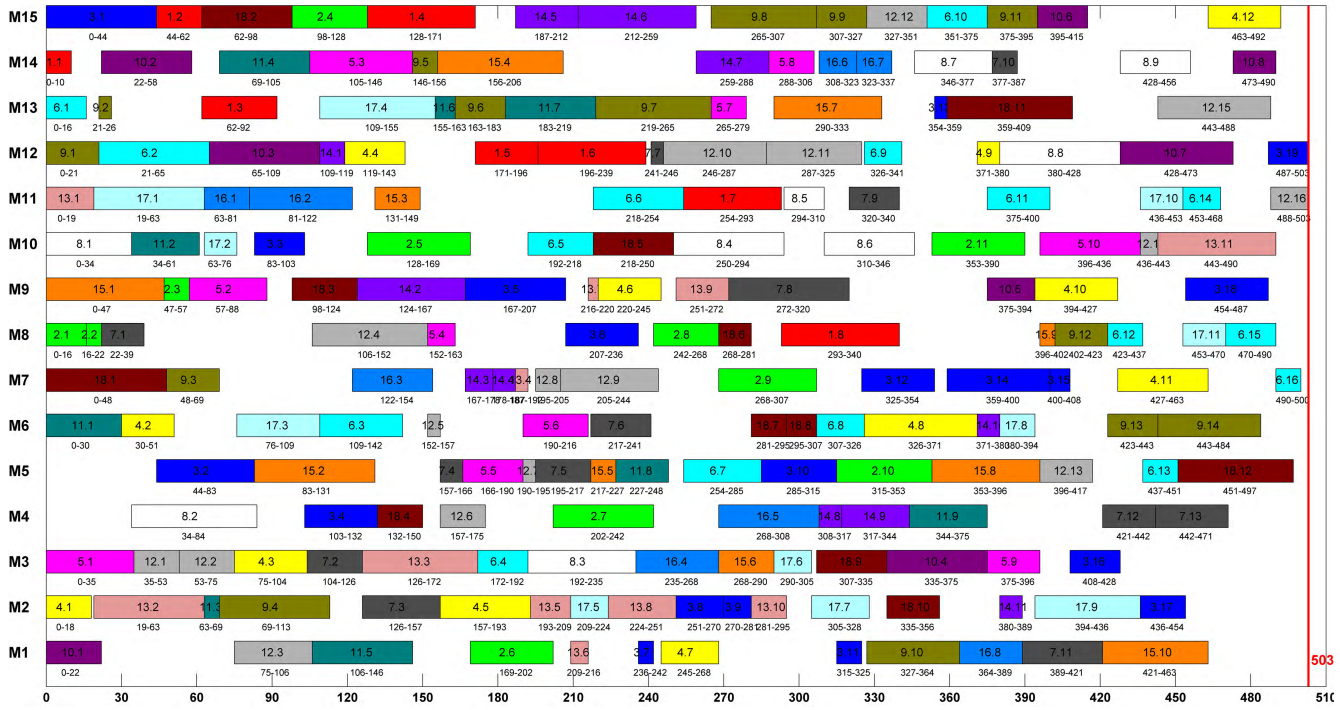


FIGURE 7. The Gantt chart of Instance 24 with nominal processing times.

For the most complex instance, e.g. Instance 24, the maximum computational time is about 1000 seconds; that is, it takes about 1000 seconds to complete 1000 iterations. However, the actual computational time needed is usually less than 1000 seconds because a solution usually converges before the 1000th iteration. For a medium or small scale instance, there is a significant decrease in computational time.

Figure 7 gives the Gantt chart of Instance 24 with nominal processing times and the nominal makespan is 503; Figure 8 presents the Gantt chart of the same instance with actual processing times. In this case, the actual makespan is 497.77. However, in other cases, the actual makespan values may be larger than the nominal one.

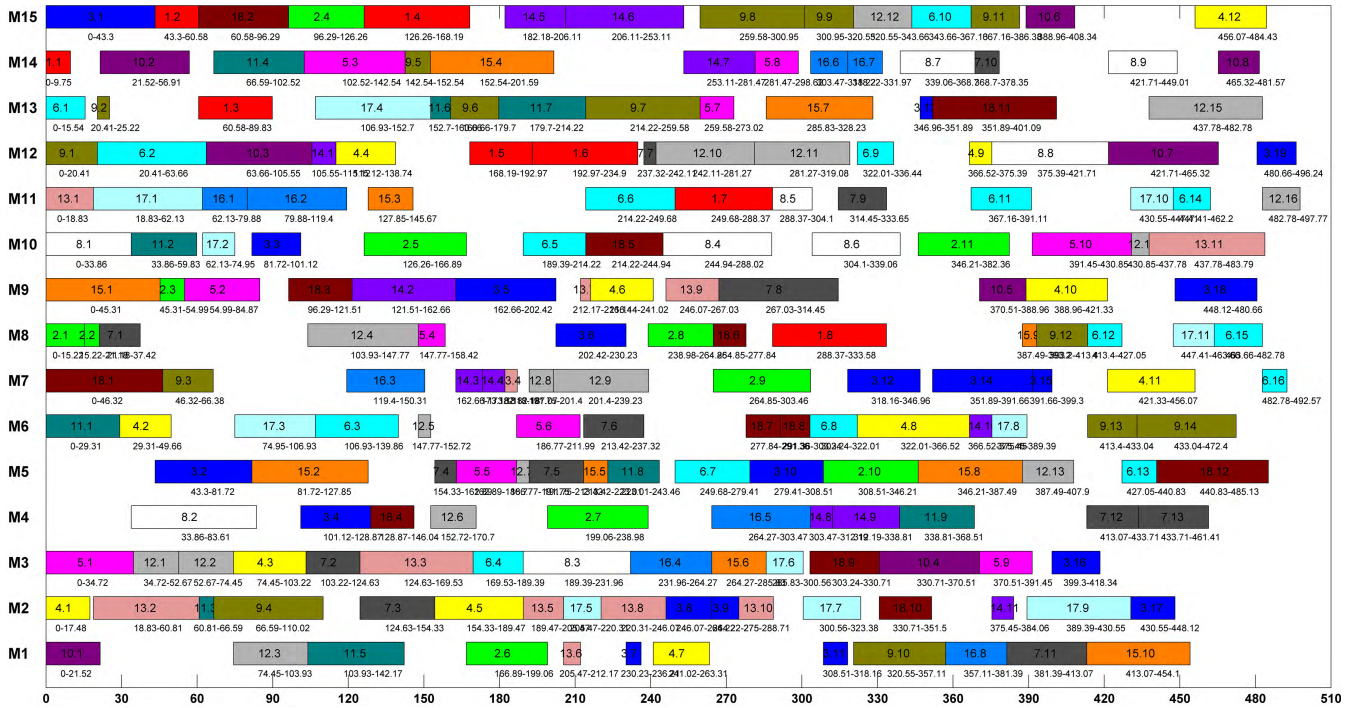


FIGURE 8. The Gantt chart of Instance 24 with actual processing times.

VI. CONCLUSIONS

The integration of process planning and scheduling can bring convenience in pursuing an efficient utilization of manufacturing resources and reducing conflicts of equipments. However, uncertain processing times in real life usually make a so-called optimal scheduling scheme inefficient or useless. Therefore, the optimal solution obtained in a deterministic environment may not perform well in real life situations and the solution method for more robust scheduling schemes is needed urgently. This paper presents a neotrosophic number based modeling technique as well as a VNS based solution method for the IPPS problem to seek more robust solutions. Based on the theory of neutrosophic numbers, corresponding Type-2 MILP model is developed to facilitate the IPPS problem with uncertain processing times. By far, it is the first MILP model for such problem. In the proposed model, the operation flexibility, the sequencing flexibility and the processing flexibility are handled properly by introducing suitable constraints or variables and more importantly, relative neutrosophic variables and some constraints are mapped into the determinacy and the indeterminacy parts. Due to the NP-hardness of the problem and the complexity in solving the problem using the proposed MILP model, a VNS based memetic algorithm is proposed for the problem to seek more robust solutions. The nominal makespan criterion and the deviation criterion, which stands for the robustness of a solution, have been considered in a weighted sum manner in the proposed memetic algorithm. Corresponding genetic operators, such as the decoding method, the selection method and the solution updating method, are also developed. The well-known Kim's benchmark instances are introduced in the experiments to test the performance of the proposed algorithm and also seeking robust solutions. Different weight factors as well as degrees of fluctuations are also defined in experiments. It shows that the solutions yielded by the proposed memetic algorithm are in general better than the ones obtained by the plain genetic algorithms where the VNS local search procedure is not considered. This reflects that the VNS local search method has powerful search capability to capture promising results. Further study indicates that the two criteria, e.g. the nominal makespan criterion and the robustness criterion are two conflict objectives and it is quite necessary to take the two criteria into account simultaneously to strike a medium between the two objectives. By setting distinct weight factor, solutions with different preferences can be obtained.

In most cases, instead of the precise probability distribution, workshop staff can only tell the range of the processing time of an operation processed by a certain machine; such characteristics of real life processing times of operations render the application of neutrosophic numbers in processing time modelling; the proposed memetic algorithm where neutrosophic numbers are used to model actual processing times can be applied to obtain more robust scheduling solutions. This research therefore presents a novel perspective as well as an optimization methodology for uncertain IPPS or other similar scheduling problems.

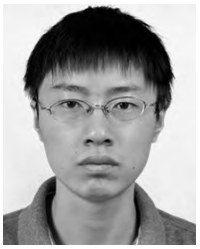
The weighted sum method is adopted in multi-objective optimization process and the resultant optimal solution is totally dependent on the weight factor w; nevertheless, such method narrows the scope of the solution space and ignores the potential satisfactory solutions. Therefore, the Pareto

based multi-objective optimization method as well as the novel neutrosophic sets based decision method [75] can be considered in further research. In addition, there are other uncertain events in the shop floor, such as job random arrivals and machine breakdowns; such uncertain events can be considered in the IPPS problem as new research directions. The IPPS problem with time-of-use (TOU) electricity price is more close to real life situations, and such research can also be performed in subsequent studies.

REFERENCES

[1] L. Jin and C. Zhang, "Process planning optimization with energy consumption reduction from a novel perspective: Mathematical modeling and a dynamic programming-like heuristic algorithm," *IEEE Access*, vol. 7, pp. 7381–7396, 2019.
[2] H. Qin, P. Fan, H. Tang, P. Huang, B. Fang, and S. Pan, "An effective hybrid discrete grey wolf optimizer for the casting production scheduling problem with multi-objective and multi-constraint," *Comput. Ind. Eng.*, vol. 128, pp. 458–476, Feb. 2019.
[3] M. Dai, D. Tang, Y. Xu, and W. Li, "Energy-aware integrated process planning and scheduling for job shops," *Proc. Inst. Mech. Engineers, Part B: J. Eng. Manuf.*, vol. 229, no. 1, pp. 13–26, Feb. 2015.
[4] X. Li, S. Xiao, C. Wang, and J. Yi, "Mathematical modeling and a discrete artificial bee colony algorithm for the welding shop scheduling problem," *Memetic Comput.*, vol. 11, no. 4, pp. 371–389, Dec. 2019.
[5] L. Zhang and T. N. Wong, "An object-coding genetic algorithm for integrated process planning and scheduling," *Eur. J. Oper. Res.*, vol. 244, no. 2, pp. 434–444, Jul. 2015.
[6] M. Haddadzade, M. R. Razfar, and M. H. F. Zarandi, "Integration of process planning and job shop scheduling with stochastic processing time," *Int. J. Adv. Manuf. Technol.*, vol. 71, nos. 1–4, pp. 241–252, Mar. 2014.
[7] T.-K. Liu, Y.-P. Chen, and J.-H. Chou, "Solving distributed and flexible job-shop scheduling problems for a real-world fastener manufacturer," *IEEE Access*, vol. 2, pp. 1598–1606, 2014.
[8] K. Lian, C. Zhang, L. Gao, and X. Li, "Integrated process planning and scheduling using an imperialist competitive algorithm," *Int. J. Prod. Res.*, vol. 50, no. 15, pp. 4326–4343, Aug. 2012.
[9] X. Li, X. Shao, L. Gao, and W. Qian, "An effective hybrid algorithm for integrated process planning and scheduling," *Int. J. Prod. Econ.*, vol. 126, no. 2, pp. 289–298, Aug. 2010.
[10] M. Kumar and S. Rajotia, "Integration of process planning and scheduling in a job shop environment," *Int. J. Adv. Manuf. Technol.*, vol. 28, nos. 1–2, pp. 109–116, Feb. 2006.
[11] O. Sobeyko and L. Mönch, "Integrated process planning and scheduling for large-scale flexible job shops using metaheuristics," *Int. J. Prod. Res.*, vol. 55, no. 2, pp. 392–409, Jan. 2017.
[12] X. Li, L. Gao, and W. Li, "Application of game theory based hybrid algorithm for multi-objective integrated process planning and scheduling," *Expert Syst. Appl.*, vol. 39, no. 1, pp. 288–297, Jan. 2012.
[13] Y. K. Kim, K. Park, and J. Ko, "A symbiotic evolutionary algorithm for the integration of process planning and job shop scheduling," *Comput. Oper. Res.*, vol. 30, no. 8, pp. 1151–1171, Jul. 2003.
[14] L. Jin, C. Zhang, X. Shao, and X. Yang, "A study on the impact of periodic and event-driven rescheduling on a manufacturing system: An integrated process planning and scheduling case," *Proc. Inst. Mech. Eng., B, J. Eng. Manuf.*, vol. 231, no. 3, pp. 490–504, Feb. 2017.
[15] L. Jin, Q. Tang, C. Zhang, X. Shao, and G. Tian, "More MILP models for integrated process planning and scheduling," *Int. J. Prod. Res.*, vol. 54, no. 14, pp. 4387–4402, Jul. 2016.
[16] C. Pan, Y. Qiao, N. Wu, and M. Zhou, "A novel algorithm for wafer sojourn time analysis of single-arm cluster tools with wafer residency time constraints and activity time variation," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 5, pp. 805–818, May 2015.
[17] B. J. Joo, S.-O. Shim, T. J. Chua, and T. X. Cai, "Multi-level job scheduling under processing time uncertainty," *Comput. Ind. Eng.*, vol. 120, pp. 480–487, Jun. 2018.
[18] L. Jin, C. Zhang, X. Shao, and G. Tian, "Mathematical modeling and a memetic algorithm for the integration of process planning and scheduling considering uncertain processing times," *Proc. Inst. Mech. Eng., B, J. Eng. Manuf.*, vol. 230, no. 7, pp. 1272–1283, Jul. 2016.
[19] V. K. Manupati, G. D. Putnik, M. K. Tiwari, P. Ávila, and M. M. Cruz-Cunha, "Integration of process planning and scheduling using mobile-agent based approach in a networked manufacturing environment," *Comput. Ind. Eng.*, vol. 94, pp. 63–73, Apr. 2016.
[20] P. Kouvelis, R. L. Daniels, and G. Vairaktarakis, "Robust scheduling of a two-machine flow shop with uncertain processing times," *IIE Trans.*, vol. 32, no. 5, pp. 421–432, May 2000.
[21] Z. Li and M. Ierapetritou, "Process scheduling under uncertainty: Review and challenges," *Comput. Chem. Eng.*, vol. 32, nos. 4–5, pp. 715–727, Apr. 2008.
[22] F. Smarandache, "Neutrosophy: Neutrosophic probability, set, and logic: Analytic synthesis & synthetic analysis," Tech. Rep, 1998.
[23] F. Smarandache, *Introduction to Neutrosophic Measure, Neutrosophic Integral, and Neutrosophic Probability*. Craiova, Romania: Sitech and Education Publisher, 2013.
[24] J. Ye, "Bidirectional projection method for multiple attribute group decision making with neutrosophic numbers," *Neural Comput. Appl.*, vol. 28, no. 5, pp. 1021–1029, May 2017.
[25] J. Chen and J. Ye, "A projection model of neutrosophic numbers for multiple attribute decision making of clay-brick selection," *Neutrosophic Sets Syst.*, vol. 12, pp. 139–142, 2016.
[26] J. Ye, "Multiple-attribute group decision-making method under a neutrosophic number environment," *J. Intell. Syst.*, vol. 25, no. 3, pp. 377–386, Jan. 2016.
[27] Z. Lu and J. Ye, "Exponential operations and an aggregation method for single-valued neutrosophic numbers in decision making," *Information*, vol. 8, no. 2, p. 62, Jun. 2017.
[28] Z. Fang and J. Ye, "Multiple attribute group decision-making method based on linguistic neutrosophic numbers," *Symmetry*, vol. 9, no. 7, p. 111, Jul. 2017.
[29] W. Jiang and J. Ye, "Optimal design of truss structures using a neutrosophic number optimization model under an indeterminate environment," *Neutrosophic Sets Syst*, vol. 14, pp. 93–97, 2016.
[30] J. Ye, "Fault diagnoses of steam turbine using the exponential similarity measure of neutrosophic numbers," *J. Intell. Fuzzy Syst.*, vol. 30, no. 4, pp. 1927–1934, Mar. 2016.
[31] J. Gao, L. Sun, and M. Gen, "A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems," *Comput. Oper. Res.*, vol. 35, no. 9, pp. 2892–2907, Sep. 2008.
[32] E. Nowicki and C. Smutnicki, "A fast taboo search algorithm for the job shop problem," *Manage. Sci.*, vol. 42, no. 6, pp. 797–813, Jun. 1996.
[33] M. Mastrolilli and L. M. Gambardella, "Effective neighbourhood functions for the flexible job shop problem," *J. Scheduling*, vol. 3, no. 1, pp. 3–20, Jan. 2000.
[34] E. Balas and A. Vazacopoulos, "Guided local search with shifting bottleneck for job shop scheduling," *Manage. Sci.*, vol. 44, no. 2, pp. 262–275, Feb. 1998.
[35] C. Y. Zhang, P. Li, Y. Rao, and Z. Guan, "A very fast TS/SA algorithm for the job shop scheduling problem," *Comput. Oper. Res.*, vol. 35, no. 1, pp. 282–294, Jan. 2008.
[36] C. Zhang, P. Li, Z. Guan, and Y. Rao, "A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem," *Comput. Oper. Res.*, vol. 34, no. 11, pp. 3229–3242, Nov. 2007.
[37] B. Liu, L. Wang, and Y.-H. Jin, "An effective PSO-based memetic algorithm for flow shop scheduling," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 37, no. 1, pp. 18–27, Feb. 2007.
[38] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," in *Technical Report Caltech Concurrent Computation Program 826*. Pasadena, CA, USA: California Institute of Technology, 1989. [Online]. Available: https://www.researchgate.net/publication/2354457_On_Evolution_Search_Optimization_Genetic_Algorithms_and_Martial_Arts_-_Towards_Memetic_Algorithms/citations
[39] P. Hansen and N. Mladenović, "Variable neighborhood search: Principles and applications," *Eur. J. Oper. Res.*, vol. 130, no. 3, pp. 449–467, May 2001.
[40] M. Yazdani, M. Amiri, and M. Zandieh, "Flexible job-shop scheduling with parallel variable neighborhood search algorithm," *Expert Syst. Appl.*, vol. 37, no. 1, pp. 678–687, Jan. 2010.
[41] M. Rainer-Harbach, P. Papazek, B. Hu, and G. R. Raidl, "Balancing bicycle sharing systems: A variable neighborhood search approach," in *Evolutionary Computation in Combinatorial Optimization*, M. Middendorf and C. Blum, eds. Berlin, Germany: Springer, 2013, pp. 121–132.

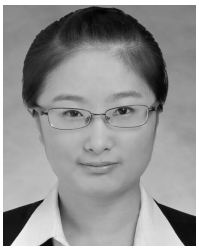
[42] O. Bräysy, "A reactive variable neighborhood search for the vehicle-routing problem with time windows," *INFORMS J. Comput.*, vol. 15, no. 4, pp. 347–368, Nov. 2003.
[43] L. Jin, C. Zhang, and X. Shao, "An effective hybrid honey bee mating optimization algorithm for integrated process planning and scheduling problems," *Int. J. Adv. Manuf. Technol.*, vol. 80, nos. 5–8, pp. 1253–1264, Sep. 2015.
[44] L. Jin, C. Zhang, X. Shao, X. Yang, and G. Tian, "A multi-objective memetic algorithm for integrated process planning and scheduling," *Int. J. Adv. Manuf. Technol.*, vol. 85, nos. 5–8, pp. 1513–1528, Jul. 2016.
[45] S. Lv and L. Qiao, "A cross-entropy-based approach for the optimization of flexible process planning," *Int. J. Adv. Manuf. Technol.*, vol. 68, nos. 9–12, pp. 2099–2110, Oct. 2013.
[46] X. Shao, X. Li, L. Gao, and C. Zhang, "Integration of process planning and scheduling—A modified genetic algorithm-based approach," *Comput. Oper. Res.*, vol. 36, no. 6, pp. 2082–2096, 2009.
[47] S. Zhang and T. N. Wong, "Integrated process planning and scheduling: An enhanced ant colony optimization heuristic with parameter tuning," *J. Intell. Manuf.*, vol. 29, no. 3, pp. 585–601, Mar. 2018.
[48] G. Luo, X. Wen, H. Li, W. Ming, and G. Xie, "An effective multi-objective genetic algorithm based on immune principle and external archive for multi-objective integrated process planning and scheduling," *Int. J. Adv. Manuf. Technol.*, vol. 91, nos. 9–12, pp. 3145–3158, Aug. 2017.
[49] M. Petrović, N. Vuković, M. Mitić, and Z. Miljković, "Integration of process planning and scheduling using chaotic particle swarm optimization algorithm," *Expert Syst. Appl.*, vol. 64, pp. 569–588, Dec. 2016.
[50] L. Jin, C. Zhang, and X. Fei, "Realizing energy savings in integrated process planning and scheduling," *Processes*, vol. 7, no. 3, p. 120, Feb. 2019.
[51] A. Seker, S. Erol, and R. Botsali, "A neuro-fuzzy model for a new hybrid integrated process planning and scheduling system," *Expert Syst. Appl.*, vol. 40, no. 13, pp. 5341–5351, Oct. 2013.
[52] U. Özcan, "Balancing stochastic two-sided assembly lines: A chance-constrained, piecewise-linear, mixed integer program and a simulated annealing algorithm," *Eur. J. Oper. Res.*, vol. 205, no. 1, pp. 81–97, Aug. 2010.
[53] M. Davari and E. Demeulemeester, "A novel branch-and-bound algorithm for the chance-constrained resource-constrained project scheduling problem," *Int. J. Prod. Res.*, vol. 57, no. 4, pp. 1265–1282, Feb. 2019.
[54] K. Ağpak and H. Gökçen, "A chance-constrained approach to stochastic line balancing problem," *Eur. J. Oper. Res.*, vol. 180, no. 3, pp. 1098–1115, Aug. 2007.
[55] A. Elyasi and N. Salmasi, "Stochastic scheduling with minimizing the number of tardy jobs using chance constrained programming," *Math. Comput. Model.*, vol. 57, nos. 5–6, pp. 1154–1164, Mar. 2013.
[56] M. Liu, X. Liu, F. Chu, F. Zheng, and C. Chu, "Profit-oriented distributionally robust chance constrained flowshop scheduling considering credit risk," *Int. J. Prod. Res.*, vol. 58, no. 8, pp. 2527–2549, Jan. 2020. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/00207543.2020.1711982
[57] B. Zhou, G. Chen, Q. Song, and Z. Y. Dong, "Robust chance-constrained programming approach for the planning of fast-charging stations in electrified transportation networks," *Appl. Energy*, vol. 262, Mar. 2020, Art. no. 114480.
[58] C. Zhang, S. Guo, F. Zhang, B. A. Engel, and P. Guo, "Towards sustainable water resources planning and pollution control: Inexact joint-probabilistic double-sided stochastic chance-constrained programming model," *Sci. Total Environ.*, vol. 657, pp. 73–86, Mar. 2019.
[59] G. Tian, M. Zhou, and J. Chu, "A chance constrained programming approach to determine the optimal disassembly sequence," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 4, pp. 1004–1013, Oct. 2013.
[60] L. Wang, G. Zhou, Y. Xu, and M. Liu, "A hybrid artificial bee colony algorithm for the fuzzy flexible job-shop scheduling problem," *Int. J. Prod. Res.*, vol. 51, no. 12, pp. 3593–3608, Jun. 2013.
[61] D. Lei, "A genetic algorithm for flexible job shop scheduling with fuzzy processing time," *Int. J. Prod. Res.*, vol. 48, no. 10, pp. 2995–3013, May 2010.
[62] M. Sakawa and T. Mori, "An efficient genetic algorithm for job-shop scheduling problems with fuzzy processing time and fuzzy duedate," *Comput. Ind. Eng.*, vol. 36, no. 2, pp. 325–341, Apr. 1999.
[63] S. Wang, L. Wang, Y. Xu, and M. Liu, "An effective estimation of distribution algorithm for the flexible job-shop scheduling problem with fuzzy processing time," *Int. J. Prod. Res.*, vol. 51, no. 12, pp. 3778–3793, Jun. 2013.
[64] K. Z. Gao, P. N. Suganthan, Q. K. Pan, M. F. Tasgetiren, and A. Sadollah, "Artificial bee colony algorithm for scheduling and rescheduling fuzzy flexible job shop problem with new job insertion," *Knowl.-Based Syst.*, vol. 109, pp. 1–16, Oct. 2016.
[65] T. Jamrus, C.-F. Chien, M. Gen, and K. Sethanan, "Hybrid particle swarm optimization combined with genetic operators for flexible job-shop scheduling under uncertain processing time for semiconductor manufacturing," *IEEE Trans. Semicond. Manuf.*, vol. 31, no. 1, pp. 32–41, Feb. 2018.
[66] S.-C. Horng, S.-S. Lin, and F.-Y. Yang, "Evolutionary algorithm for stochastic job shop scheduling with random processing time," *Expert Syst. Appl.*, vol. 39, no. 3, pp. 3603–3610, Feb. 2012.
[67] K. Wang and S. Choi, "A holonic approach to flexible flow shop scheduling under stochastic processing times," *Comput. Oper. Res.*, vol. 43, pp. 157–168, Mar. 2014.
[68] C. Liu, Q. Zeng, H. Duan, M. Zhou, F. Lu, and J. Cheng, "E-net modeling and analysis of emergency response processes constrained by resources and uncertain durations," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 1, pp. 84–96, Jan. 2015.
[69] Y. W. Guo, W. D. Li, A. R. Mileham, and G. W. Owen, "Applications of particle swarm optimisation in integrated process planning and scheduling," *Robot. Comput.-Integr. Manuf.*, vol. 25, no. 2, pp. 280–288, Apr. 2009.
[70] Z. S. Xu and Q. L. Da, "The uncertain OWA operator," *Int. J. Intell. Syst.*, vol. 17, no. 6, pp. 569–575, Jun. 2002.
[71] C. Özgüven, L. Özbakür, and Y. Yavuz, "Mathematical models for job-shop scheduling problems with routing and process plan flexibility," *Appl. Math. Model.*, vol. 34, no. 6, pp. 1539–1548, Jun. 2010.
[72] P. M. França, A. Mendes, and P. Moscato, "A memetic algorithm for the total tardiness single machine scheduling problem," *Eur. J. Oper. Res.*, vol. 132, no. 1, pp. 224–242, Jul. 2001.
[73] H.-E. Tseng, "Guided genetic algorithms for solving a larger constraint assembly problem," *Int. J. Prod. Res.*, vol. 44, no. 3, pp. 601–625, Feb. 2006.
[74] F. Zhang, Y. F. Zhang, and A. Y. C. Nee, "Using genetic algorithms in process planning for job shop machining," *IEEE Trans. Evol. Comput.*, vol. 1, no. 4, pp. 278–289, Nov. 1997.
[75] J. Ye, "Single valued neutrosophic cross-entropy for multicriteria decision making problems," *Appl. Math. Model.*, vol. 38, no. 3, pp. 1170–1175, Feb. 2014.



LIANGLIANG JIN received the Ph.D. degree in mechatronic engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2016. He is currently a Lecturer of the Department of Mechanical Engineering, Shaoxing University. He has authored or coauthored over several academic articles. His research interests mainly focused on modeling and optimization of production manufacturing systems for the efficient and effective utilization of resources, such as the integrated process planning and scheduling problem and the flexible job shop scheduling problem, and sustainable manufacturing, including cleaner production and green manufacturing. He is also a person in charge of a national scientific research project and a provincial scientific research project.

CHAOYONG ZHANG (Member, IEEE) received the B.S. degree in mechanical engineering from the Tianjin University of Science and Technology, Tianjin, China, in 1993, the M.S. degree in mechatronic engineering from the University of Science and Technology Beijing, Beijing, China, in 1999, and the Ph.D. degree in mechatronic engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2007.

He is currently an Associate Professor with the School of Mechanical Science and Engineering, HUST. He has authored two books, and authored or coauthored over 70 academic articles. His research interests mainly focus on modeling and optimization of production manufacturing systems, including (flexible) job shop scheduling optimization, the assembly line balancing problem, and the integrated process planning and scheduling problem. He has been honored by the Ministry of Education Natural Science First Prize, in 2013, the China Machinery Industry Federation First Prize, in 2010, and the Ministry of Education Science and Technology Progress First Prize, in 2008.



XIAOYU WEN received the Ph.D. degree in industrial engineering from the Huazhong University of Science and Technology, in 2014. She is currently a Lecturer with the School of Mechanical and Electrical Engineering Institute, Zhengzhou University of Light Industry. She has authored or coauthored over 30 academic articles. Her research interests mainly focused on integrated process planning and scheduling, multi-objective optimization, intelligent optimization algorithm, and digital twin. She has been honored by the Henan Provincial Science and Technology Progress Award Third Prize, in 2017.

GEORGE GERSHOM CHRISTOPHER was born in Port Harcourt, Nigeria, in 1996. He received the B.Sc. degree (Hons.) in computer engineering from ISFOP-BENIN UNIVERSITY (Autonomous), Cotonou, Benin, in 2016. He is currently pursuing the M.Sc. degree in civil engineering, with specialization in project management, with Shaoxing University, China. He has more than five years in telecommunication industry experience, work on big data analysis, computer maintenance, including working for Ama computer limited (Zinox computers) and Transsnet Limited. He is also working on two researches on the civil engineering scheduling problem with uncertain processing times and the Internet of Things, with a focus on social impact. His current research interests include big data analysis and geographical information systems, artificial intelligent, computer system engineering, and cyber security.