A modified integer and categorical PSO algorithm for solving integrated process planning, dynamic scheduling and due date assignment problem

Caner Erden\textsuperscript{a}, Halil İbrahim Demir\textsuperscript{b}, and Onur Canpolat\textsuperscript{b,}\textsuperscript{*}

\textit{a. Department of International Trade and Finance, Faculty of Applied Science, Sakarya University of Applied Science, Sakarya, Turkey}
\textit{b. Department of Industrial Engineering, Faculty of Engineering, Sakarya University, Turkey,}

+90 264 6160427 (+905547793441), +90 264 295 5675 (+905363886015), +90 264 295 7442 (+905321797124)

\texttt{cerden@subu.edu.tr, hidemir@sakarya.edu.tr, mailto:onurcanpolat@sakarya.edu.tr}

Abstract: Particle Swarm Optimization (PSO) has many successful applications in solving continuous optimization problems. It has been adapted to solve discrete optimization problems using different variants, such as integer PSO (IPSO), discrete PSO (DPSO) and integer and categorical PSO (ICPSO). ICPSO, a recent PSO variant, uses probability distributions instead of the solution values. In this study, we applied ICPSO algorithm to solve dynamic integrated process planning, scheduling and due date assignment (DIPPSDDA) problem which is a higher integration level of well-known problems which are integrated process planning and scheduling (IPPS) and scheduling with due date assignment (SWDDA). Briefly, due date assignment function is integrated to IPPS problem as the third manufacturing function in DIPPSDDA. Furthermore, DIPPSDDA performs scheduling function in a dynamic environment where jobs arrive at the shop floor at any time. The objective of DIPPSDDA problem is to minimize the earliness, tardiness and given due dates length. Since the experimental results show that ICPSO converges, crossover and mutation operators used in genetic algorithms were implemented to ICPSO, namely modified ICPSO (MICPSO). Finally, experimental results indicate that the proposed MICPSO provides better performance as compared to genetic algorithms, ICPSO and modified discrete PSO.
Keywords: Dynamic Scheduling, Dynamic Scheduling and Due Date Assignment, Integer and Categorical PSO, Genetic Algorithms, Integrated Process Planning Scheduling and Due Date Assignment

1. Introduction

Process planning, scheduling and due date assignment functions are three important functions for manufacturing environments. The first function, process planning prepares an engineering design into the final product by preparing detailed processing instructions and determines the necessary resources, machines and routes to produce a product [1]. The second function, scheduling is a decision-making process that takes the time-to-task assignment of scarce resources and aims to optimize one or more purposes [2]. The third function, due date assignment is a concept which increases its importance especially after the spread of the just-in-time concept and aims to deliver products to customers at the right time. In traditional manufacturing environments, these functions are usually handled separately that may cause inefficient schedules and due dates. Also, process plans which are prepared independently provide poor inputs to the scheduling as illustrated in Fig. 1.

Studies on the integration of these three functions have been done rarely. In addition, common due date is determined in most of the studies and customers' weights are not considered. Weights of the customers are crucial for businesses because it is undesirable to process low-priority jobs earlier from others or to assign a very long due date time to very important customers.

Integrated Process Planning and Scheduling (IPPS) is a well-known study area in the literature. IPPS has the benefits of choosing an alternative route, finding solutions for urgent jobs, and balancing the load of machines. IPPS increases the production efficiency of the companies, also helps to meet the demands on time and optimizes utilization of processes and resources. Especially when the studies in the last decade are examined,
Zhang and Wong [3] used the ant colony algorithm from heuristic approaches to solve the IPPS problem in a job shop environment. Sobeyko and Mönch [4] have implemented an IPPS application on a large scale flexible job shop type production environment where different product trees and routes can be found. They have addressed the weighted total delay as the objective function. They provided mixed integer programming for this problem. Chaudhry [5] proposed a genetic algorithm (GA) based on Microsoft Excel for the IPPS problem in the job shop environment. Luo et al. [6] addressed the multi-objective integrated process planning and scheduling (MOIPPS) problem, in that a production system required multiple objectives to be taken into account in the more realistic decision-making process. Zhang and Wong [7] studied three different models of IPPS problems, including setup times. They used a customized ant colony algorithm. Petrovic et al. [8] have tried a new heuristic antlion optimization for IPPS problem and showed its applicability. Manupati et al. [9] discussed a mobile-agent-based approach for IPPS. Besides, some of the studies on this subject can be given as Meenakshi Sundaram and Fu [10], Khoshnevis and Chen [11], Zhang and Mallur [12], Morad and Zalzala [13], Phanden et al. [14], Li and Gao [15], Phanden et al. [16], Li et al. [17] and Lin et al. [18].

There are also many studies in the literature on scheduling with due date assignment (SWDDA). Some of the studies are carried out in single machine environments, while others are performed in multi-machine environments. In most of the studies in the literature, it is seen that due dates are determined with respect to process times and the number of operations. Chen [19] and Gordon et al. [20] are further references on SWDDA. In recent years Zhao et al. [21], Xiong et al. [22], Yin et al. [23], Liu et al. [24], Wang et al. [25] Yin et al. [26] and Wang et al. [27] have studied on this area. Zhao et al. [21] examined a single machine scheduling and due date assignment, where the processing time of a job depends on both the start time and the position in the queue. Xiong et al. [22] discussed the problem of single machine scheduling and due date assignment in an environment
where machines are disrupted by a certain probability randomly. They have aimed to minimize the optimal job sequence and costs while setting the common due date. Wang et al. [25] investigated the multi-agent single-machine SWDDA problem, each aimed at optimizing its own performance. Shabtay [28] has investigated a scheduling problem in a single machine environment where due dates are controllable with batch delivery. Yin et al. [29] carried out a single machine SWDDA study including the costs of delivering jobs.

Increasing the integration of these functions will help to improve the scheduling performance and global production efficiency of a manufacturing system. When these three functions are integrated, there will be strong communication between each other and provide better inputs to one another. Fig. 2 shows the benefits of the integrated manufacturing functions. When the studies that integrate three functions (process planning, scheduling and due date assignment - IPPSDDA) are examined, there are limited studies in the literature. Yuan [30] proved that only the IPPS problem to minimize early and tardy jobs and batch delivery costs is NP-hard. Thus, solving the problem of integration of the three functions will be even more complex. Demir and Taskin [31] have studied IPPSDDA as their Ph.D. thesis. Then, Ceven and Demir [32] studied on benefits of integrating the due date assignment with IPPS problem and Demir and Erden [33] studied on the integration of three functions by genetic algorithm and ant colony optimization to minimize the sum of weighted earliness, tardiness and due-dates of every job. Demir and Phanden [34] reviewed IPPSDDA literature in the book edited by Phanden et al. [16]. The IPPSDDA problem is a workable and remarkable research area.

This study deals with the dynamic events on a shop floor. Some internal or external dynamic events may occur in real shop floors. For example, a machine break-down, an urgent job, or changes in due dates can lead to a breakdown of the previously prepared schedules or the occurrence of ineffective schedules. To overcome these problems, schedules should be made to react to dynamic events. Thus, the dynamic scheduling
approach, which is prepared using real-time information and adapted to unexpected events, will provide more successful results [35]. When the studies on dynamic scheduling are examined, it is revealed that dynamic scheduling is harder to solve than solving static scheduling [35], [36] and [37]. In this study, new job arrivals were considered as dynamic events and process planning, scheduling and due date assignment are integrated.

There are numerous methods to solve combinational problems. One of them is PSO which is a heuristic algorithm that is often used as it is easy to implement in solving optimization problems. PSO has been widely used in literature, especially in IPPS problems. Ba et al. [38], have developed a multimodal program using the PSO algorithm for mass production to minimize production time. Petrovic et al. [39] focused on using a method based on PSO algorithm and chaos theory to investigate the field of search more extensively in IPPS problems and avoid the local optima. Yu et al. [40] developed a hybrid algorithm based on GA and PSO to solve the IPPS problem, which includes two different phases, static and dynamic. Petrovic et al. [41] offered a new algorithm for the optimization of flexible process plans based on the use of PSO and chaos theory. Wang et al. [42] also tried to solve the multipurpose IPPS problem by using PSO. In addition to these studies, many PSO studies are included in the literature. However, almost all PSO studies have variables that are of continuous value. This indicates that the classical PSO will be insufficient to solve discrete optimization models. As a result of the studies, many PSO variations such as integer PSO (IPSO), discrete PSO (DPSO), Binary PSO, Veeramachaneni PSO, Angle Modulated PSO, Discrete Estimation of Distribution Particle Swarm Optimization (DEDPSO) have been developed. One of the PSO variations that are used to eliminate this deficiency is the integer and categorical PSO (ICPSO), which is a type of PSO in which the values of the particles in the swarm are expressed by probability distributions. ICPSO has been used to solve DIPPSDDA problem in this study, since the due date assignment rules, the dispatching rules and the routes of the jobs have categorical data characteristics. In
most scheduling studies, the makespan is used as an objective function. However, in this study, it is aimed to minimize the total weighted earliness, total weighted tardiness and given weighted due dates length (E/T/D).

With all our knowledge, the studies on DIPPSDDA are quite limited in the literature. Erden et al. [43] studied DIPPSDDA problem for the first time. In their study, genetic algorithms, simulated annealing, taboo search, and combination of these algorithms were used to solve the problem. Later Demir and Erden [33] studied DIPPSDDA problem by using ACO Algorithm. In this study, ICPSO algorithm will be used for the first time for solving DIPPSDDA problem. The developed algorithm is modified, and it has been observed that the modified algorithm gives better results.

The remainder of this paper is organized as follows. The DIPPSDDA problem is discussed in Section 2. PSO variants (ICPSO, MICPSO, MDPSO) are mentioned in Section 3. The developed algorithm steps are given in Section 4. Comparative results of the algorithm are given in Section 5. Finally, the results of the study and future works are mentioned in Section 6.

2. Problem Definition

DIPPSDDA problem can be considered as a contribution to the dynamic job shop scheduling problem (DJSSP). In general, in the job shop scheduling problem (JSSP), n jobs are assigned to m machine regarding an objective function. There are some assumptions in JSSP such as each machine can only perform one job at the same time, one operation can be processed on one machine at the same time, the machine does not fail when the operation starts on the machine, no urgent job arrives at the shop floor etc. On the contrary, machines may be broken down or urgent jobs may arrive at the shop floor in the dynamic JSSP model. Since the problem of DIPPSDDA is a variant of dynamic JSSP.
problem, it is assumed that an urgent job can arrive at the shop floor. Besides, alternative process plans are used to create more effective schedules.

In a DIPPSDDA model, \( n \) number of jobs with \( o \) number of operations and \( r \) number of different routes are processed through \( m \) number of machines. The processing time of an operation on a machine is generated using a normal distribution with a mean of 12 and a standard deviation of 6. Jobs are arriving at the system according to an exponential distribution with a mean of 10 and associated with a due date which is calculated using a due date assignment rule. Operation pending machine queue is selected by machine using dispatching rules as well. In this study, 8 different sizes of shop floors (SF) are produced for the problem and the data of the shop floors is shared in Table 1.

As mentioned earlier, in an IPPS model, scheduling is carried out considering all process plans of the jobs. This enables more efficient and balanced scheduling for the shop floor. Another function integrated with scheduling is the due date assignment function. Significant gains in production efficiency can be achieved with the proper time of due dates.

Scheduling problems may have a static or dynamic nature. In dynamic scheduling, unexpected events, such as machine break downs, new job arrivals or changes in due dates may affect the performance of the existing schedules. Unexpected events in a shop floor may result in loss of optimal schedules or they may generate infeasible schedules. To handle these problems, it is important to consider the unexpected events while scheduling. Dynamic scheduling models are much closer to actual shop floors and they are the most difficult problems to be solved among the scheduling problems. In this study, new job arrivals are studied as dynamic events.
The objective function developed for the problem is given as minimizing total weighted earliness, tardiness, and due dates’ length of every job. Tardiness and earliness are calculated as in Eq. (1-2):

\[ T_j = \max(c_j - d_j, 0) \]  
\[ E_j = \max(d_j - c_j, 0) \]

Where \( T_j \), \( E_j \) denote the tardiness and earliness time of the \( j \)-th job, respectively. \( c_j \), \( d_j \) corresponds to the completion and given due date time of the \( j \)-th job, respectively.

If the job is completed after its given due date time, tardiness will occur. In case of tardiness, the penalty for early completion is 0, as expected. If the job is completed before its given due date time, earliness will occur and in case of the early completion time, the tardiness penalty is given as 0. Weighted due dates are penalized along with weighted earliness and tardiness and the penalty values of \( E/T/D \) are calculated as in Eq. (3-6):

\[ P_D = w_j \times \left( 8 \times \frac{d_j}{480} \right) \]  
\[ P_E = w_j \times \left( 5 + 4 \times \frac{E_j}{480} \right) \]  
\[ P_T = w_j \times \left( 8 + 6 \times \frac{T_j}{480} \right) \]  
\[ P_{total} = P_D + P_E + P_T \]

Where \( P_D, P_E, P_T, P_{total} \) denote the penalties of a due date, earliness, tardiness, and total penalty, respectively. \( w_j \) is the weight of the \( j \)-th job. The objective function of the model is to minimize the total penalties. Then the final objective function \((f_{sol})\) which is a fitness value of the solution is represented as in Eq. (7):

\[ f_{sol} = \text{total } P \]
Many due date assignment rules have been developed in studies [25], [44], [45], [46] and [47]. It can be experimentally revealed which of these rules will give better results. The due date assignment rules used in this study are given in Table 2 with explanation and equations.

Where \( a_i, p_i, o_i \) denote the arrival time, processing time and the operation number of the \( i \)-th job, respectively. \( P_w \) is the average processing time of all waiting jobs. \((w_i, w_o)\) are determined proportionally inverse to the job weights and \( k_i = 1, 2, 3 \) and

\[
q_1 : q_1 = \frac{P_w}{2}, q_2 = P_w, q_3 = 3\frac{P_w}{2}.
\]

After determining the due dates of each job, dispatching rule for scheduling must be determined. There are also many dispatching rule studies in the literature. With these rules, it is determined which of the waiting job is to be processed next. Dispatching rules can be divided into 4 categories which are process time based, due date based, combination rules, and mixed-based rules [48]. For example, the SPT rule is a process-based rule. In SPT, the job with the shortest processing time is prioritized. Process time-based rules do not take due dates into account. EDD can be given as an example of rules that consider the due date [36]. The EDD rule prioritizes the job with the shortest due date. In the combination rules, the slack or critical rate is determined. Detailed studies on this subject can be given as follows: Adibi et al. [49], Amin and El-Bouri [50], Dominic et al. [51], Heger et al. [52], Pierreval and Mebarki [53], Qi et al. [54], Baker and Kanet [55], Raghu and Rajendran [56], Vepsalainen and Morton [57]. As in the rules for due date assignment, it is possible to find out which dispatching rule gives better solutions as a result of experimental studies. The formulas of the priority index of dispatching rules are given in Table 3.
Where \( I_i \) denotes the priority index of the \( i \)-th job and \( \max I_i \) is selected among the jobs waiting. Slack is calculated as given Eq. (8):

\[
\text{slack} = d_i - p_i - a_i
\]

(8)

3. Application of ICPSO to DIPPSDDA

Proposed ICPSO algorithm is based on the study of Strasser et al. [58]. When the algorithm is applied with Strasser’s form, solution performance got stuck to local minima. To overcome this, ICPSO has been modified by the addition of crossover and mutation operators. Thus, the developed algorithm has been made more useful for integrated scheduling problems. The position vector of particles in the classical PSO structure is denoted as \( X_p \), which is a candidate solution of particle \( p \). To keep categorical data in ICPSO, a particle representation is created using probability distributions. All dimensions in this representation create probability distributions for a solution to the problem. For this integrated problem, \( X_p \) is divided into 3 parts \( X_p = [X_{p, \text{ddrule}}, X_{p, \text{dsprule}}, X_{p, \text{routes}}] \) as each part is valid at different intervals. For \( p \)-th particle, its due date assignment rule position can be represented as \( X_{p, \text{ddrule}} = [D_{p,1, \text{ddrule}}, D_{p,2, \text{ddrule}}, \ldots, D_{p, N1, \text{ddrule}}] \) where each \( D_{p,i, \text{ddrule}} \) represents the probability distribution for variable \( X_{p,i, \text{ddrule}} \) and \( N1 \) denotes the due date assignment rule size. Then, every element in the particle’s due date position vector is also consists of a set of distributions \( D_{p,i, \text{ddrule}} = [d_{p,i,1, \text{ddrule}}, d_{p,i,2, \text{ddrule}}, \ldots, d_{p,i,N1, \text{ddrule}}] \) where \( d_{p,i, \text{ddrule}} \) denotes the probability variable \( X_{p,i, \text{ddrule}} \) assumes value \( a \) for \( p \)-th particle. Similarly, for \( p \)-th particle, the dispatching rule position vector can be represented as \( X_{p, \text{dsprule}} = [D_{p,1, \text{dsprule}}, D_{p,2, \text{dsprule}}, \ldots, D_{p, N2, \text{dsprule}}] \) where \( N2 \) denotes dispatching rule size. For routes dimension, we need probability distributions for all jobs in the shop floor which can be represented as
$X_{p,\text{routes}} = \left[ \begin{array}{c} D^1_{p,\text{routes}} \\ D^2_{p,\text{routes}} \\ \vdots \\ D^N_{p,\text{routes}} \end{array} \right]$ where $D^{j-k}_{p,\text{routes}}$ denotes the probability distribution of $X^{j-k}_{p,\text{routes}}$ for the routes of $k$-th job and $p$-th particle and $N$ is the number of routes. The probability vector routes of each job can be represented as $D^{j-k}_{p,\text{routes}} = [d^{1,j-k}_{p,\text{routes}}, d^{2,j-k}_{p,\text{routes}}, \ldots, d^{N,j-k}_{p,\text{routes}}]$ where $d^{a,j-k}_{p,\text{routes}}$ denotes the probability that variable $X^{j-k}_{p,\text{routes}}$ assumes value $a$ of the $p$-th particle.

In classical PSO, particles move using velocity vectors. Again, we have 3 dimensions for velocity vectors. A particle’s due date assignment rule vector of $n$ vectors $\phi$ which is one for each variable in the solution. The velocity vector of the due date assignment rules dimension can be represented as $V_{p,\text{ddrule}} = \left[ \phi^1_{p,\text{ddrule}}, \phi^2_{p,\text{ddrule}}, \ldots, \phi^N_{p,\text{ddrule}} \right]$ and $\phi^i_{p,\text{ddrule}} = \left[ \psi^i_{p,\text{ddrule}}, \psi^2_{p,\text{ddrule}}, \ldots, \psi^N_{p,\text{ddrule}} \right]$ where $\psi^i_{p,\text{ddrule}}$ denotes $p$-th particle velocity for $i$-th due date assignment rule. Similarly, dispatching rules can be represented as $V_{p,\text{dsprule}} = \left[ \phi^1_{p,\text{dsprule}}, \phi^2_{p,\text{dsprule}}, \ldots, \phi^N_{p,\text{dsprule}} \right]$ and $\phi^i_{p,\text{dsprule}} = \left[ \psi^i_{p,\text{dsprule}}, \psi^2_{p,\text{dsprule}}, \ldots, \psi^N_{p,\text{dsprule}} \right]$.

Lastly, routes vector consists of $n$ jobs can be represented as $V_{p,\text{routes}} = \left[ V^{j-1}_{p,\text{routes}}, V^{j-2}_{p,\text{routes}}, \ldots, V^{j-N}_{p,\text{routes}} \right]$ and routes of $k$-th job can be represented as $V^{j-k}_{p,\text{routes}} = \left[ \phi^{j-k}_{p,\text{routes}}, \phi^{j-k}_{p,\text{routes}}, \ldots, \phi^{N,j-k}_{p,\text{routes}} \right]$ where $\phi^{j,k}_{p,\text{routes}} = \left[ \psi^{j-k}_{p,\text{routes}}, \psi^{j-k}_{p,\text{routes}}, \ldots, \psi^{N,j-k}_{p,\text{routes}} \right]$.

The velocity vector in the classical PSO has been modified to make it effective for this problem specifically. The particles update position vectors at each iteration. The velocity vector of due date assignment rule, dispatching rule and routes of each job are given in Eq. (9-11):

\begin{align*}
V_{p,\text{ddrule}} &= \omega V_{p,\text{ddrule}} + c_1 r_1 \left( p_{\text{best \; p,\text{ddrule}}} + X_{p,\text{ddrule}} \right) + c_2 r_2 \left( p_{\text{\text{gbest \; p,\text{ddrule}}} + X_{p,\text{ddrule}}} \right) \\
V_{p,\text{dsprule}} &= \omega V_{p,\text{dsprule}} + c_1 r_1 \left( p_{\text{best \; p,\text{dsprule}}} + X_{p,\text{dsprule}} \right) + c_2 r_2 \left( p_{\text{\text{gbest \; p,\text{dsprule}}} + X_{p,\text{dsprule}}} \right) \\
V_{p,\text{routes}} &= \omega V_{p,\text{routes}} + c_1 r_1 \left( p_{\text{best \; p,\text{routes}}} + X_{p,\text{routes}} \right) + c_2 r_2 \left( p_{\text{\text{gbest \; p,\text{routes}}} + X_{p,\text{routes}}} \right)
\end{align*}

Where;
\( \omega \): inertia rate,

\( c_1 \): cognition constant,

\( r_1 \): cognition random number within the range of \([0,1] \)

\( c_2 \): social constant

\( r_2 \): social random number within the range of \([0,1]\)

\( p_{best} \): Global best particle

\( p_{best} \): Personal best particle

New position vectors that are created from velocity vectors are given in Eq. (12-14):

\[
X_{p,\text{ddrule}} = X_{p,\text{ddrule}} + V_{p,\text{ddrule}} \\
X_{p,\text{drule}} = X_{p,\text{drule}} + V_{p,\text{drule}} \\
X_{j-k,\text{p,routes}} = X_{j-k,\text{p,routes}} + V_{j-k,\text{p,routes}}
\]  

Where \( X_{p,\text{routes}} = [X_{j=1,\text{p,routes}}, X_{j=2,\text{p,routes}}, \ldots, X_{j=n,\text{p,routes}}] \). The proposed method is presented as follows: firstly, an initial swarm with random probabilities is generated. Then, the particle’s due date assignment, dispatching rules and routes of each job values are determined according to probabilities. All solution values in the initial swarm are also recorded as personal best value \((p_{p\text{best}})\) of the particles. Thus, the initial swarm is obtained. The fitness value of the particle which has the best fitness value in the swarm is recorded as global best values \((p_{g\text{best}})\). At the same time, the best particle is saved as a particle of \(p_{g\text{best}}\).

In the next iterations, the probabilities for the particles of the swarm are updated using PSO velocity and position equations. The value of the due date assignment and dispatching rules and routes obtained according to the assigned probability values and the fitness for the particle is calculated. If the fitness value \((F)\) of the current particle’s \((p_{\text{current}})\) is better than the current particle’s best fitness \((p_{\text{best}})\), the \(p_{\text{best}}\) of the particle is
updated and the particle is recorded as $p_{best}$. Then, the same process is done for all particles, it is examined whether there is a particle with a value of better fitness than $p_{best}$. If a better fitness value is achieved, the $p_{best}$ is updated and the relevant particle is saved as $p_{best}$ particle. The algorithm is iterated until the iteration size $k_{iter,dl}$. Pseudo codes of the proposed ICPSO is given in Algorithm 1.

4. Other solution approaches

4.1. Genetic Algorithms (GA)

Genetic algorithms were developed by Holland [59]. GA is focused on solving computational optimization problems, inspired by the evolutions of species. Iterations are performed based on the high probability of individuals with better compliance values in the GA to move to the next population. Iterations re-selection, mutation and crossover operators are used. GA has solved many scheduling and IPPS problems optimization. Similar studies with the problem discussed in this study can be given as follows: Li et al. [60], Lin et al. [61], Park and Choi [62], Pezzella et al. [63], Xia et al. [64], Zhang et al. [65].

The proposed algorithm is working with several steps. In the initialize population step, we proposed a classical GA for solving DIPPSDDA problem as well. Each gene of the solution chromosome stores a due date assignment rule, a dispatching rule, and routes of jobs, respectively. For initialization of GA, a random search is applied for 20 iterations and the 10 best chromosomes are selected for the initial population.

Selection: GA selection operator selects 3 pairs of chromosomes and 4 chromosomes for crossover and mutation operations, respectively. A ranking probability method is applied for the selection operator which has better performance for this problem. Because the performance difference between the best and the worst chromosome is getting smaller
towards the end of the iterations. Chromosome probabilities for the selection operator are fixed at every iteration and are given in Table 4. In here, we give more probability to be selected to the chromosomes which have better fitness values.

Crossover: Firstly, we determine the crossover point number which is based on the number of jobs as the number of jobs is related to the chromosome size. The number of crossover points is calculated using \( \text{ceil}(\text{gene}_{\text{size}} \times 0.1) \) equation. Secondly, we select crossover points with the selection probabilities of each gene. In here, we have two dominant genes which are due date assignment and dispatching rule genes. Probabilities of those genes are given 0.25, 0.25, respectively. Other genes which are the selected routes of the jobs are given 0.5 probabilities in total. Because due date assignment and dispatching rule genes have a significant impact on the fitness value. If we change these genes, the performance function will be dramatically affected. On the other hand, if we change a route of a job, performance function will not be affected that much. These genes should be selected more to see which pair working well together. Therefore, these two genes have been identified as dominant genes in the solution and a higher selection probability has been given those genes. Thirdly, we apply a multi-point crossover operator between two parent chromosomes to produce new two offspring chromosomes.

Mutation: Like the crossover operator, we determine the mutation points number using the following equation \( \text{ceil}(\text{gene}_{\text{size}} \times 0.3) \) in the first step. At the second step, we apply the mutation operator to the selected genes. After crossover and mutation operators, we have a new population with 20 chromosomes. To fix the population size to 10, we determined the population by selecting the best 10 chromosomes. At the last step, we iterate the selection, crossover and mutation operators until the iteration number is reached. The parameters setting of GA is given in Table 4.

4.2. Modified ICPSO (GA/ICPSO)
We applied different variants of PSO to find better solutions. In the algorithm of MICPSO, we add mutation and crossover operators to classical ICPSO. ICPSO tries to optimize the probabilities of the solution but in this study when pure ICPSO is used improvements become harder. That is why we integrate crossover and mutation operators to the ICPSO algorithm and change a certain number of genes at inertia, cognitive and social part of the algorithm. ICPSO calculates the probabilities and generates new solution values for the problem at each iteration using \( \mathbf{V} \) and \( \mathbf{X} \) vectors. Applying mutation operator and then crossover to each particle with \( p_{best} \) and \( g_{best} \) improve solution performance. At first, we applied mutation operator to each particle and with 0.25 probability each gene is replaced with possible other values and this part constitutes the moment of inertia part of the algorithm. Later, each particle is crossed over with \( p_{best} \) and each gene is replaced with 0.25 probability with the associated \( p_{best} \) values and this part constitutes the cognitive part of the algorithm. Finally, with 0.25 probability each gene is changed over with associated \( g_{best} \) values and this constitutes the social part of the algorithm. These updates are all applied for all particles in the swarm and we obtained better results. These steps are stopped when the iteration size is reached. Pseudo codes of the proposed MICPSO is given in Algorithm 2.

4.3. Modified Discrete PSO (MDPSO)

Here, Modified Discrete Particle Swarm Optimization (MDPSO) [66] is adapted to solve the problem in which the possible structure of ICPSO has been tried to be improved. In the study of Pan et al. [66], a probability is given for every gene of particles to be mutated or crossed over. For example, according to Pan et al. [66], firstly every gene is mutated with approximately 25% probability and later every gene is changed into associated \( p_{best} \) value with 25% probability and finally ever gene is changed into associated \( g_{best} \) value with 25% probability.
In our MDPSO algorithm, we select %25 of genes of the particle randomly. Those genes are mutated and changed into another possible value (Moment of Inertia applied). After that, we select again randomly %25 of genes of the particle and these genes are changed into associated $p_{best}$ values (Cognitive part is applied). Thus, each particle and associated $p_{best}$ values are crossed over. Finally, we select %25 of genes of the particle again and change the values of these genes with associated $g_{best}$ values (Social part is applied) and thus, the crossover is applied between every particle and $g_{best}$ particle. This makes MDPSO more practical to apply. Pseudo codes of the proposed MDPSO is given in Algorithm 3.

5. Experimental Results

In this study, the proposed algorithms are coded in Python programming language on Intel® Core™ i5-6200U CPU @ 2.30GHz with 8 GB RAM a personal computer. Appropriate Python packages such as NumPy [67], Matplotlib [68], pandas [69], salabim [70] are utilized to analyze and solve the problem. Events such as new job arrivals, the end of an operation of the jobs or the assignment of a job to a machine are simulated with the help of salabim package. Besides, the job to be selected by the machine among the jobs waiting for machine queue is made by taking into consideration the dispatching rule. Thus, it is aimed to find the optimal dispatching rule, due date assignment rule and routes of each job combinations. Because there is no research data on DIPPSDDA in published papers, we generated 8 different sizes of shop floors and their data for this problem. The data used for this study is given as the supplementary file.

One of the outcomes of this study is the most appropriate schedules for production. Schedules obtained from the last iteration can be shown using Gantt charts. Gantt charts created for this study also shows the arrival of jobs. The arrival time of the jobs and the first machine to be assigned at the time of arrival are shown with the help of arrows. A Gantt chart is created for all shop floors, but only a Gantt chart is shown for the first shop
floor. As it is too hard to follow charts in medium and big size shops. Each job in the diagram is shown in a different color. Boxed pieces show the operations of jobs. Since there are 10 operations in every job, the jobs are shown with 10 pieces. The Gantt chart of the optimal schedule obtained by MICPSO is shown in Fig. 3.

The proposed MICPSO algorithm is applied to the data and the experimental results of MICPSO compared with the results of MDPSO, ICPSO and GA in 8 different sizes of shop floors which are illustrated in Fig. 4.

As it can be seen from Fig. 4, MICPSO gave the best results in all shop floors except Shop Floor 1, 4 and 5. From figures, mostly MICPSO algorithm has better performance than the other algorithms. Meanwhile, the CPU time of ICPSO is better than other algorithms. Also, the best, average and worst results for all shop floors can be seen in detail in Fig. 5.

According to Table 5, MICPSO and MDPSO outperformed GA and pure ICPSO. At the five out of eight shop floors, MICPSO gave better performance but in shop floor 1, 4 and 5 MDPSO gave better performance. Since MDPSO is very practical to apply, it is also a promising solution technique but mostly MICPSO gave better performance and can be recommended for DIPPSDDA problems.

According to Table 6, most of the jobs have earliness. This is because tardiness is undesired with greater fixed and variable cost terms. Fixed and variable cost parameters for earliness are 5 and 4, but on the other hand, fixed and variable cost parameters and coefficients for tardiness are 8 and 6. If a job is tardy instead of early than we penalize the job with an additional 3-unit fixed penalty in terms of fixed cost and variable cost coefficient becomes 6 instead of 4. The last column of the table shows the total penalty
for every job and if we sum up the last column then we get the total penalty of all jobs which gives fitness function for this shop floor.

The best, average (Avg) and worst results of executing eight shop floors with all algorithms are presented in Table 7. In general, MICPSO had better performance for eight shop floors by having minimum best values mostly. Further analysis about the performance of the algorithms was done using one-way analysis of variance (ANOVA) test to check if there is a significant difference between the results of the algorithms. Average values of the fitness functions are selected as the response values and the algorithms are assumed as factors. To perform the ANOVA analysis, we run the program ten times with different seed values in the shop floor 8 where the highest variability is expected. The results are given in Table 8.

Before performing ANOVA test, we need to check the normality assumption. As a result of the normality test, it was determined that normality is not satisfied as it can be seen in Fig. 6 ($p < 0.010$). For this reason, the non-parametric test, Kruskal-Wallis test, was performed instead of ANOVA. Kruskal-Wallis test results show a strong significant difference between the algorithm results because the $p$-value (0.007) is too close to zero as shown in Table 9 and 10. The means plot in the least significant difference intervals at 99% confidence is illustrated in Fig. 7. Those results are indicating the superiority of the MICPSO algorithm compared with GA, MDPSO and ICPSO.

6. Conclusion

In this study, process planning, dynamic scheduling and due date assignment functions are integrated which is a novel subject in the literature. It is assumed that the jobs arrive at the shop floor with the exponential distribution randomly. The problem is modelled, and popular population-based PSO and GA algorithms are preferred from meta-heuristic
algorithms as solution methods. Since GA solution has been introduced in the previous studies [71,72], the structure of PSO which is developed and modified for the solution of the problem is mentioned more than GA in the application section of the paper. The results of the experimental studies show that MICPSO has better performance and quality and is one of the best methods in terms of both the best solution and CPU usage rates. Since classical PSO is usually worked with continuous data, ICPSO, a variation of PSO, was utilized in this study due to the discrete and categorical nature of the problem. It has been ensured that ICPSO is modified for the problem with some improvements. Since ICPSO is a newly developed PSO variation algorithm; the implementation of the algorithm among the NP-hard combinational problems is limited in the literature. Only scheduling problem with more than 3 machines is an NP-hard optimization problem [73]. Here a study has been carried out to fulfil this gap. The method developed has been called modified PSO and saved as a new method for further studies. To sum up, these conditions indicate the original aspects of the study.

With DIPPSDDA more efficient, effective, and balanced schedules in the shop floors can be obtained. Because process plans, schedules and due dates are tried to be optimized using the alternative process plans in DIPPSDDA. IPPS problems have reached a certain number of studies and many issues have been studied so far. Therefore, there is a need for new study subjects and ideas. We presented a new study area for the researchers working on IPPS and SWDDA. This issue needs further work in the future. We can list the possible future research that can be focused on as follows:

Comparison of the other discrete methods with proposed ICPSO.

- Solving DIPPSDDA model with other successful algorithms (Artificial Bee Colony, Honeybee Colony etc.)
- Including more objectives such as makespan to consider the DIPPSDDA problem in the form of target programming.
• Adding other dynamic events to the DIPPSDDA problem such as machine breakdowns, job cancellations.

• Integrating other production functions such as delivery manufacturing function to the DIPPSDDA problem.

Supplementary Materials: Representative data in the main text added here as supplementary “dataset.zip” zip file.

7. References

1. Chang, T. C. and Wysk, R. A. An Introduction to Automated Process Planning Systems, Prentice Hall Professional Technical Reference (1984).

2. Kahlbacher, H. G. and Cheng, T. C. E. “Parallel machine scheduling to minimize costs for earliness and number of tardy jobs”, Discrete Appl. Math., 47(2), pp. 139–164 (1993).

3. Zhang, S. and Wong, T. N. “Integrated process planning and scheduling: an enhanced ant colony optimization heuristic with parameter tuning”, J. Intell. Manuf., 29(3), pp. 585–601 (2018).

4. Sobeyko, O. and Mönch, L. “Integrated process planning and scheduling for large-scale flexible job shops using metaheuristics”, Int. J. Prod. Res., 55(2), pp. 392–409 (2017).

5. Chaudhry, I. A. “A Genetic Algorithm Approach for Process Planning and Scheduling in Job Shop Environment”, p. 6 (2012).

6. Luo, G., Wen, X., Li, H., et al. “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., 91(9), pp. 3145–3158 (2017).

7. Zhang, S. and Wong, T. N. “Studying the impact of sequence-dependent set-up times in integrated process planning and scheduling with E-ACO heuristic”, Int. J. Prod. Res., 54(16), pp. 4815–4838 (2016).

8. Petrović, M., Petronijević, J., Mitić, M., et al. “The Ant Lion Optimization Algorithm for Integrated Process Planning and Scheduling”, https://www.scielo.org/AMM.834.187.
9. Manupati, V. K., Putnik, G. D., Tiwari, M. K., et al. “Integration of process planning and scheduling using mobile-agent based approach in a networked manufacturing environment”, *Comput. Ind. Eng.*, **94**, pp. 63–73 (2016).

10. Meenakshi Sundaram, R. and Fu, S. “Process planning and scheduling — A method of integration for productivity improvement”, *Comput. Ind. Eng.*, **15**(1–4), pp. 296–301 (1988).

11. Khoshnevis, B. and Chen, Q. M. “Integration of process planning and scheduling functions”, *J. Intell. Manuf.*, **2**(3), pp. 165–175 (1991).

12. Zhang, H. C. and Mallur, S. “An integrated model of process planning and production scheduling”, *Int. J. Comput. Integr. Manuf.*, **7**(6), pp. 356–364 (1994).

13. Morad, N. and Zalzala, A. “Genetic algorithms in integrated process planning and scheduling”, *J. Intell. Manuf.*, **10**(2), pp. 169–179 (1999).

14. Phanden, R. K., Jain, A., and Verma, R. “Integration of process planning and scheduling: a state-of-the-art review”, *Int. J. Comput. Integr. Manuf.*, **24**(6), pp. 517–534 (2011).

15. Li, X. and Gao, L. *Effective Methods for Integrated Process Planning and Scheduling*, Engineering Applications of Computational Methods, Springer-Verlag, Berlin Heidelberg (2020).

16. Phanden, R. K., Jain, A., and Davim, J. P., Eds. *Integration of Process Planning and Scheduling: Approaches and Algorithms*, 1st edition, CRC Press, Boca Raton (2019).

17. Li, X., Gao, L., Pan, Q., et al. “An Effective Hybrid Genetic Algorithm and Variable Neighborhood Search for Integrated Process Planning and Scheduling in a Packaging Machine Workshop”, *IEEE Trans. Syst. Man Cybern. Syst.*, **49**(10), pp. 1933–1945 (2019).

18. Lin, C. S., Li, P. Y., Wei, J. M., et al. “Integration of process planning and scheduling for distributed flexible job shops”, *Comput. Oper. Res.*, **124**, p. 105053 (2020).

19. Chen, Z. L. “Scheduling and common due date assignment with earliness-tardiness penalties and batch delivery costs”, *Eur. J. Oper. Res.*, **93**(1), pp. 49–60 (1996).

20. Gordon, V., Proth, J. M., and Chu, C. “A survey of the state-of-the-art of common due date assignment and scheduling research”, *Eur. J. Oper. Res.*, **139**(1), pp. 1–25 (2002).

21. Zhao, C., Hsu, C. J., Lin, W. C., et al. “Due date assignment and scheduling with time and positional dependent effects”, *J. Inf. Optim. Sci.*, pp. 1–14 (2018).
22. Xiong, X., Wang, D., Cheng, T. C. E., et al. “Single-machine scheduling and common due date assignment with potential machine disruption”, *Int. J. Prod. Res.*, **56**(3), pp. 1345–1360 (2018).

23. Yin, Y., Wang, W., Wang, D., et al. “Multi-agent single-machine scheduling and unrestricted due date assignment with a fixed machine unavailability interval”, *Comput. Ind. Eng.*, **111**, pp. 202–215 (2017).

24. Liu, W., Hu, X., and Wang, X. “Single machine scheduling with slack due dates assignment”, *Eng. Optim.*, **49**(4), pp. 709–717 (2017).

25. Wang, D. J., Yin, Y., Cheng, S. R., et al. “Due date assignment and scheduling on a single machine with two competing agents”, *Int. J. Prod. Res.*, **54**(4), pp. 1152–1169 (2016).

26. Yin, Y., Wang, D., and Cheng, T. C. E. *Due Date-Related Scheduling with Two Agents : Models and Algorithms*, Uncertainty and Operations Research, Springer Singapore (2020).

27. Wang, Y., Wang, J. Q., and Yin, Y. “Multitasking scheduling and due date assignment with deterioration effect and efficiency promotion”, *Comput. Ind. Eng.*, **146**, p. 106569 (2020).

28. Shabtay, D. “Scheduling and due date assignment to minimize earliness, tardiness, holding, due date assignment and batch delivery costs”, *Int. J. Prod. Econ.*, **123**(1), pp. 235–242 (2010).

29. Yin, Y., Cheng, T. C. E., Wu, C. C., et al. “Single-machine batch delivery scheduling and common due-date assignment with a rate-modifying activity”, *Int. J. Prod. Res.*, **52**(19), pp. 5583–5596 (2014).

30. Yuan, J. “A note on the complexity of single-machine scheduling with a common due date, earliness-tardiness, and batch delivery costs”, *Eur. J. Oper. Res.*, **94**(1), pp. 203–205 (1996).

31. Demir, H.İ. and Taskin, H. “Integrated Process Planning, Scheduling and Due-Date Assignment”, PhD Thesis, Sakarya University (2005).

32. Ceven, E. and Demir, H.İ. “Benefits of Integrating Due-Date Assignment with Process Planning and Scheduling”, Master of Science Thesis, Sakarya University (2007).

33. Demir, H. İ. and Erden, C. “Dynamic integrated process planning, scheduling and due-date assignment using ant colony optimization”, *Comput. Ind. Eng.*, **149**, p. 106799 (2020).
34. Demir, H. İ. and Phanden, R. K. *Due-Date Agreement in Integrated Process Planning and Scheduling Environment Using Common Meta-Heuristics*, CRC Press (2019).

35. Ouelhadj, D. and Petrovic, S. “A survey of dynamic scheduling in manufacturing systems”, *J. Sched.*, **12**(4), pp. 417–431 (2009).

36. Ramasesh, R. “Dynamic job shop scheduling: A survey of simulation research”, *Omega*, **18**(1), pp. 43–57 (1990).

37. Yin, L., Gao, L., Li, X., et al. “An improved genetic algorithm with rolling window technology for dynamic integrated process planning and scheduling problem”, 2017 *IEEE 21st Int. Conf. Comput. Support. Coop. Work Des. CSCWD*, pp. 414–419 (2017).

38. Ba, L., Li, Y., Yang, M., et al. “A Mathematical Model for Multiworkshop IPPS Problem in Batch Production”, *Math. Probl. Eng.*, **2018**, p. 7948693 (2018).

39. Petrović, M., Vuković, N., Mitić, M., et al. “Integration of process planning and scheduling using chaotic particle swarm optimization algorithm”, *Expert Syst. Appl.*, **64**, pp. 569–588 (2016).

40. Yu, M., Zhang, Y., Chen, K., et al. “Integration of process planning and scheduling using a hybrid GA/PSO algorithm”, *Int. J. Adv. Manuf. Technol.*, **78**(1), pp. 583–592 (2015).

41. Petrović, M., Mitić, M., Vuković, N., et al. “Chaotic particle swarm optimization algorithm for flexible process planning”, *Int. J. Adv. Manuf. Technol.*, **85**(9), pp. 2535–2555 (2016).

42. Wang, Y. F., Zhang, Y. F., and Fuh, J. Y. H. “A PSO-based multi-objective optimization approach to the integration of process planning and scheduling”, *IEEE ICCA 2010*, pp. 614–619 (2010).

43. Erden, C., Demir, H. İ., and Kökçam, A. H. “Solving Integrated Process Planning, Dynamic Scheduling, and Due Date Assignment Using Metaheuristic Algorithms”, *Math. Probl. Eng.*, **2019**, p. 1572614 (2019).

44. Janiak, A., Janiak, W. A., Krysiak, T., et al. “A survey on scheduling problems with due windows”, *Eur. J. Oper. Res.*, **242**(2), pp. 347–357 (2015).

45. Yin, Y., Wang, D. J., Wu, C. C., et al. “CON/SLK due date assignment and scheduling on a single machine with two agents”, *Nav. Res. Logist. NRL*, **63**(5), pp. 416–429 (2016).
46. Browning, T. R. and Yassine, A. A. “Resource-constrained multi-project scheduling: Priority rule performance revisited”, *Int. J. Prod. Econ.*, **126**(2), pp. 212–228 (2010).

47. Sha, D. Y. and Liu, C. H. “Using Data Mining for Due Date Assignment in a Dynamic Job Shop Environment”, *Int. J. Adv. Manuf. Technol.*, **25**(11), pp. 1164–1174 (2005).

48. Haupt, R. “A survey of priority rule-based scheduling”, *Oper. Res. Spektrum*, **11**(1), pp. 3–16 (1989).

49. Adibi, M. A., Zandieh, M., and Amiri, M. “Multi-objective scheduling of dynamic job shop using variable neighborhood search”, *Expert Syst. Appl.*, **37**(1), pp. 282–287 (2010).

50. Amin, G. R. and El-Bouri, A. “A minimax linear programming model for dispatching rule selection”, *Comput. Ind. Eng.*, **121**, pp. 27–35 (2018).

51. Dominic, P. D. D., Kaliyamoorthy, S., and Kumar, M. S. “Efficient dispatching rules for dynamic job shop scheduling”, *Int. J. Adv. Manuf. Technol.*, **24**(1), pp. 70–75 (2004).

52. Heger, J., Branke, J., Hildebrandt, T., et al. “Dynamic adjustment of dispatching rule parameters in flow shops with sequence-dependent set-up times”, *Int. J. Prod. Res.*, **54**(22), pp. 6812–6824 (2016).

53. Pierreval, H. and Mebarki, N. “Dynamic scheduling selection of dispatching rules for manufacturing system”, *Int. J. Prod. Res.*, **35**(6), pp. 1575–1591 (1997).

54. Qi, J. G., Burns, G. R., and Harrison, D. K. “The Application of Parallel Multipopulation Genetic Algorithms to Dynamic Job-Shop Scheduling”, *Int. J. Adv. Manuf. Technol.*, **16**(8), pp. 609–615 (2000).

55. Baker, K. R. and Kanet, J. J. “Job shop scheduling with modified due dates”, *J. Oper. Manag.*, **4**(1), pp. 11–22 (1983).

56. Raghu, T. S. and Rajendran, C. “An efficient dynamic dispatching rule for scheduling in a job shop”, *Int. J. Prod. Econ.*, **32**(3), pp. 301–313 (1993).

57. Vepsalainen, A. P. J. and Morton, T. E. “Priority Rules for Job Shops with Weighted Tardiness Costs”, *Manag. Sci.*, **33**(8), pp. 1035–1047 (1987).

58. Strasser, S., Goodman, R., Sheppard, J., et al. “A New Discrete Particle Swarm Optimization Algorithm”, *Proc. 2016 Genet. Evol. Comput. Conf. - GECCO 16*, ACM Press, Denver, Colorado, USA, pp. 53–60 (2016).
59. Holland, J. H. “Genetic Algorithms”, *Sci. Am.*, 267(1), pp. 66–73 (1992).

60. Li, X., Gao, L., and Shao, X. “An active learning genetic algorithm for integrated process planning and scheduling”, *Expert Syst. Appl.*, 39(8), pp. 6683–6691 (2012).

61. Lin, S., Goodman, E. D., and Punch, W. F. “A Genetic Algorithm Approach to Dynamic Job Shop Scheduling”, *Probl. Proc. Seventh Int. Conf. Genet. Algorithms*, pp. 481–489 (1997).

62. Park, B. J. and Choi, H. R. “A Genetic Algorithm for Integration of Process Planning and Scheduling in a Job Shop”, *AI 2006 Adv. Artif. Intell.*, A. Sattar and B. Kang, Eds., Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 647–657 (2006).

63. Pezzella, F., Morganti, G., and Ciaschetti, G. “A genetic algorithm for the Flexible Job-shop Scheduling Problem”, *Comput. Oper. Res.*, 35(10), pp. 3202–3212 (2008).

64. Xia, H., Li, X., and Gao, L. “A hybrid genetic algorithm with variable neighborhood search for dynamic integrated process planning and scheduling”, *Comput. Ind. Eng.*, 102, pp. 99–112 (2016).

65. Zhang, L., Gao, L., and Li, X. “A hybrid genetic algorithm and tabu search for a multi-objective dynamic job shop scheduling problem”, *Int. J. Prod. Res.*, 51(12), pp. 3516–3531 (2013).

66. Pan, Q. K., Tasgetiren, M. F., and Liang, Y. C. “A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem”, *Comput. Oper. Res.*, 35(9), pp. 2807–2839 (2008).

67. Oliphant, T. E. *A Guide to NumPy*, Trelgol Publishing USA (2006).

68. Hunter, J. D. “Matplotlib: A 2D graphics environment”, *Comput. Sci. Eng.*, 9(3), p. 90 (2007).

69. McKinney, W. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and iPython*, 2 edition, O’Reilly Media (2017).

70. van der Ham, R. “Salabim: discrete event simulation and animation in Python”, *J. Open Source Softw.*, 3(27), p. 767 (2018).

71. Demir, H. İ., Canpolat, O., Erden, C., et al. “Process Planning and Scheduling with WNOPPT Weighted Due-Date Assignment where Earliness, Tardiness and Due-Dates are Penalized”, *J. Intell. Syst.*, p. 10 (2018).
72. Demir, H. I. and Erden, C. “Solving process planning and weighted scheduling with WNOPPT weighted due-date assignment problem using some pure and hybrid meta-heuristics”, Sak. Univ. J. Sci., 21(2), pp. 210–222 (2017).

73. Garey, M. R., Johnson, D. S., and Sethi, R. “The Complexity of Flowshop and Jobshop Scheduling”, Math. Oper. Res., 1(2), pp. 117–129 (1976).
List of Figure Caption

Figure 1. Unintegrated process planning, scheduling and due date assignment

Figure 2. Integrated process planning, scheduling and due date assignment

Figure 3. First Shop Floor Gantt Chart of the Optimal Schedule

Figure 4. Comparative results of the proposed algorithms for shop floors ((a)-Shop Floor 1, (b)-Shop Floor 2, (c)-Shop Floor 3, (d)-Shop Floor 4, (e)-Shop Floor 5, (f)-Shop Floor 6, (g)-Shop Floor 7, (h)-Shop Floor 8)

Figure 5. Comparative best, average and worst results of the proposed algorithms for shop floors

Figure 6. The normality test plot

Figure 7. Interval Plot of average results for the algorithms

List of Table Caption

Table 1. Shop floor configuration

Table 2. Due date assignment rules

Table 3. Dispatching rules

Table 4. The parameters setting of GA

Table 5. Best algorithms and fitness values for all shop floors

Table 6. Experimental results for Shop Floor 1

Table 7. Comparative results for shop floors

Table 8. Algorithm results with different seeds

Table 9. Kruskal-Wallis Descriptive Statistics
Table 10. Kruskal-Wallist test results

Algorithm 1. Pseudo code of the proposed ICPSO

Algorithm 2. MICPSO Pseudocode

Algorithm 3. MDPSO Pseudocode
Figure 1.
Figure 2.
Figure 3.
Figure 4.
Figure 5.
Figure 7.
Table 1.

| Numbers     | SF1 | SF2 | SF3 | SF4 | SF5 | SF6 | SF7 | SF8 |
|-------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Jobs        | 25  | 50  | 75  | 100 | 125 | 150 | 175 | 200 |
| Machines    | 5   | 10  | 15  | 20  | 25  | 30  | 35  | 40  |
| Operations  | 10  | 10  | 10  | 10  | 10  | 10  | 10  | 10  |
| Routes      | 5   | 5   | 5   | 5   | 3   | 3   | 3   | 3   |
| Iterations  | 150 | 150 | 100 | 100 | 75  | 75  | 50  | 50  |
Table 2.

| Rule No | Name     | Explanation                      | Equations               |
|---------|----------|----------------------------------|-------------------------|
| 0,1,2   | SLK      | Slack                            | \( d_i = a_i + p_i + q_i \) |
| 3,4,5   | WSLK     | Weighted slack                   | \( d_i = a_i + p_i + w_i q_i \) |
| 6,7,8   | TWK      | Total work content               | \( d_i = a_i + k_i p_i \) |
| 9,10,11 | WTK      | Weighted total work content      | \( d_i = a_i + w_i k_i p_i \) |
| 12,13,14| NOPPT    | Number of operations plus        | \( d_i = a_i + p_i + 5k_i a_i \) |
|         |          | processing time                  |                         |
| 15,16,17| WNOPPT   | The weighted number of           | \( d_i = a_i + p_i + 5w_i k_i a_i \) |
|         |          | operations plus processing time   |                         |
| 18      | RDM      | Random-allowance due dates       | \( d_i = a_i + N - (3P_{aw} + P_{w}) \) |
| 19,20,21,22,23,24,25,26,27 | PPW | Processing-time-plus-wait       | \( d_i = a_i + k_i p_i + q_i \) |
| 28,29,30,31,32,33,34,35,36 | WPPW | Weighted processing-time-plus-wait | \( d_i = a_i + w_2 k_i p_i + w_i q_i \) |
| Rule No | Name    | Explanation                        | Equations                                                      |
|---------|---------|------------------------------------|----------------------------------------------------------------|
| 0-1-2  | WATC    | Weighted Apparent Tardiness Cost   | \[ I_i = \frac{w_i}{p_i} e^{\frac{\max(\text{slack}, 0)}{K^p}} \] |
| 3-4-5  | ATC     | Apparent Tardiness Cost            | \[ I_i = \frac{1}{p_i} e^{\frac{\max(\text{slack}, 0)}{K^p}} \] |
| 6      | WMS     | Weighted Minimum Slack             | \[ I_i = -\left(\text{slack}\right) w_i \]                  |
| 7      | MS      | Minimum Slack                      | \[ I_i = -\left(\text{slack}\right) \]                      |
| 8      | WSPT    | Weighted shortest process time     | \[ I_i = \frac{w_i}{p_i} \]                                  |
| 9      | SPT     | Shortest process time              | \[ I_i = \frac{1}{p_i} \]                                    |
| 10     | WLPT    | Weighted longest process time      | \[ I_i = \frac{p_i}{w_i} \]                                  |
| 11     | LPT     | Longest process time               | \[ I_i = \frac{p_i}{1} \]                                    |
| 12     | WSOT    | Weighted shortest operation time   | \[ I_i = \frac{w_i}{p_i} \]                                  |
| 13     | SOT     | Shortest operation time            | \[ I_i = \frac{1}{p_i} \]                                    |
| 14     | WLOT    | Weighted longest operation time    | \[ I_i = \frac{p_i}{w_i} \]                                  |
| 15     | LOT     | Longest operation time             | \[ I_i = \frac{p_i}{1} \]                                    |
| 16     | EDD     | Earliest due date                  | \[ I_i = \frac{1}{d_i} \]                                    |
| 17     | WEDD    | Weighted Earliest due date         | \[ I_i = \frac{w_i}{d_i} \]                                  |
| 18     | ERD     | Earliest release date              | \[ I_i = \frac{1}{a_i} \]                                    |
| 19     | WERD    | Weighted earliest release date     | \[ I_i = \frac{w_i}{a_i} \]                                  |
| 20     | SIRO    | Service in random order            | \[ \text{random} \]                                          |
| 21     | FIFO    | First in first out                 | \[ I_i = \frac{1}{a_i} \]                                    |
| 22     | LIFO    | Last in first out                  | \[ I_i = a_i \]                                              |
| Parameters                              | SF1               |
|----------------------------------------|-------------------|
| Population size                        | 10                |
| Crossover probability                  | 0.7               |
| Mutation probability                   | 0.3               |
| Number of crossover points             | $\text{ceil}((n+2)*0.1)$ |
| Number of mutation points              | $\text{ceil}((n+2)*0.3)$ |
| Chromosome probabilities for selection | [0.3, 0.2, 0.15, 0.12, 0.10, 0.07, 0.03, 0.02, 0.006, 0.004] |
| Shop Floor | Best Algorithm | Fitness Value |
|------------|----------------|---------------|
| SF1        | MDPSO          | 155.93        |
| SF2        | MICPSO         | 285.09        |
| SF3        | MICPSO         | 348.69        |
| SF4        | MDPSO          | 470.66        |
| SF5        | MDPSO          | 629.16        |
| SF6        | MICPSO         | 682.71        |
| SF7        | MICPSO         | 810.50        |
| SF8        | MICPSO         | 964.31        |
| Job Weight | Arrival time(s) | Departure time(s) | Given due date(s) | Earliness | Tardiness | Penalty earliness | Penalty tardiness | Penalty due dates | Penalty total |
|------------|----------------|------------------|------------------|-----------|-----------|------------------|------------------|------------------|---------------|
| 0          | 1.00           | 71               | 218              | 274.41    | 56.41     | 0                | 5.47             | 0                | 3.39          | 8.86         |
| 1          | 0.66           | 76               | 260              | 269.28    | 9.28      | 0                | 3.35             | 0                | 2.13          | 5.48         |
| 2          | 0.33           | 83               | 227              | 268.41    | 41.41     | 0                | 1.76             | 0                | 1.02          | 2.78         |
| 3          | 0.33           | 88               | 303              | 284.66    | 0         | 18.34            | 0                | 2.06             | 1.08          | 3.14         |
| 4          | 0.66           | 98               | 290              | 291.28    | 1.28      | 0                | 3.31             | 0                | 2.13          | 5.43         |
| 5          | 1.00           | 124              | 259              | 299.28    | 40.28     | 0                | 5.34             | 0                | 2.92          | 8.26         |
| 6          | 0.33           | 133              | 312              | 321.78    | 9.78      | 0                | 1.68             | 0                | 1.04          | 2.72         |
| 7          | 0.66           | 188              | 334              | 362.16    | 28.16     | 0                | 3.45             | 0                | 1.92          | 5.37         |
| 8          | 0.66           | 196              | 339              | 362.28    | 23.28     | 0                | 3.43             | 0                | 1.83          | 5.26         |
| 9          | 0.66           | 218              | 364              | 442.78    | 78.78     | 0                | 3.73             | 0                | 2.47          | 6.21         |
| 10         | 0.33           | 288              | 432              | 493.66    | 61.66     | 0                | 1.82             | 0                | 1.13          | 2.95         |
| 11         | 0.66           | 353              | 484              | 520.41    | 36.41     | 0                | 3.5              | 0                | 1.84          | 5.34         |
| 12         | 0.66           | 367              | 646              | 554.66    | 0         | 91.34            | 0                | 4.71             | 2.06          | 6.78         |
| 13         | 1.00           | 371              | 553              | 557.53    | 4.53      | 0                | 5.04             | 0                | 3.11          | 8.15         |
| 14         | 0.66           | 377              | 619              | 550.03    | 0         | 68.97            | 0                | 4.53             | 1.9           | 6.43         |
| 15         | 1.00           | 393              | 573              | 595.28    | 22.28     | 0                | 5.19             | 0                | 3.37          | 8.56         |
| 16         | 0.66           | 408              | 556              | 590.03    | 34.03     | 0                | 3.49             | 0                | 2             | 5.49         |
| 17         | 0.66           | 416              | 753              | 616.03    | 0         | 136.97           | 0                | 5.09             | 2.2           | 7.29         |
| 18         | 1.00           | 429              | 578              | 602.03    | 24.03     | 0                | 5.20             | 0                | 2.88          | 8.08         |
| 19         | 0.66           | 458              | 601              | 628.78    | 27.78     | 0                | 3.45             | 0                | 1.88          | 5.33         |
| 20         | 1.00           | 467              | 601              | 620.91    | 19.91     | 0                | 5.17             | 0                | 2.57          | 7.73         |
| 21         | 0.33           | 468              | 647              | 705.16    | 58.16     | 0                | 1.81             | 0                | 1.3           | 3.11         |
| 22         | 1.00           | 492              | 716              | 716.78    | 0.78      | 0                | 5.01             | 0                | 3.75          | 8.75         |
| 23         | 0.66           | 496              | 657              | 664.53    | 7.53      | 0                | 3.34             | 0                | 1.85          | 5.2          |
| 24         | 1.00           | 537              | 727              | 744.91    | 17.91     | 0                | 5.15             | 0                | 3.47          | 8.61         |
|       | GA   | Avg | Worst | ICPSO | Avg | Worst | MICPSO | Avg | Worst | MDPSO | Avg | Worst |
|-------|------|-----|-------|-------|-----|-------|--------|-----|-------|-------|-----|-------|
| SF1   | 175.8| 176.4| 188.4 | 176.5 | 182.1| 204.6 | 163.2  | 166.4| 204.6 | 155.9 | 161.6| 192.3 |
| SF2   | 304.6| 308.8| 349.1 | 306.5 | 307.2| 366.5 | 285.1  | 296.3| 366.5 | 292.6 | 309.3| 350.2 |
| SF3   | 377.5| 378.5| 386.4 | 359.2 | 365.7| 414.0 | 348.7  | 357.5| 414.0 | 349.1 | 366.9| 414.1 |
| SF4   | 486.4| 503.8| 533.0 | 488.5 | 494.1| 582.0 | 473.1  | 509.7| 582.0 | 470.7 | 477.4| 499.4 |
| SF5   | 636.0| 693.3| 838.1 | 650.9 | 658.8| 630.5 | 635.9  | 658.8| 629.2 | 650.5 | 797.3|
| SF6   | 690.7| 692.7| 701.9 | 705.2 | 708.6| 809.8 | 682.7  | 739.0| 809.8 | 685.4 | 689.0| 699.7 |
| SF7   | 897.1| 898.5| 910.8 | 832.3 | 834.9| 844.3 | 810.5  | 818.6| 844.3 | 813.6 | 822.1| 859.1 |
| SF8   | 1098.3| 1099.8| 1108.6| 994.9 | 996.6| 1000.1| 994.9  | 996.6| 1000.1| 967.9 | 1047.3| 1087.0|
Table 8.

| Seeds | Best | Avg | Worst | GA   | Best | Avg | Worst | ICPSO | Best | Avg | Worst | MICPSO | Best | Avg | Worst | MDPSO | Best | Avg | Worst |
|-------|------|-----|-------|------|------|-----|-------|-------|------|-----|-------|--------|------|-----|-------|-------|------|-----|-------|
| 1     | 974.3| 991.8| 1180.1| 988.3| 998.9| 1104.7| 966.0| 982.1| 1104.7| 993.8| 1086.3| 1292.2 |
| 2     | 976.1| 985.8| 1010.2| 982.8| 1018.3| 1220.9| 978.4| 993.4| 1220.9| 973.5| 981.9| 992.9 |
| 3     | 977.0| 987.5| 1172.3| 984.6| 993.3| 1017.9| 965.1| 977.8| 1017.9| 971.0| 977.2| 1008.7|
| 4     | 992.8| 1032.2| 1068.1| 983.0| 994.3| 1076.1| 967.7| 984.6| 1076.1| 1042.6| 1051.8| 1082.0|
| 5     | 1088.7| 1104.5| 1158.4| 989.0| 1013.0| 1151.0| 974.2| 1029.7| 1151.0| 970.1| 977.3| 1005.4|
| 6     | 988.0| 1046.5| 1079.4| 986.0| 987.7| 989.3| 962.1| 969.6| 989.3| 970.2| 979.9| 1101.2|
| 7     | 1050.9| 1058.4| 1068.2| 988.1| 992.1| 999.4| 967.2| 974.2| 999.4| 970.1| 1020.9| 1110.9|
| 8     | 1088.6| 1091.4| 1106.4| 989.0| 993.0| 997.9| 963.5| 968.0| 997.9| 967.9| 1017.0| 1233.6|
| 9     | 1078.4| 1088.9| 1141.4| 990.0| 1005.8| 1221.8| 967.8| 1002.8| 1221.8| 968.9| 976.7| 1003.3|
| 10    | 1098.3| 1099.8| 1108.6| 994.9| 996.6| 1000.1| 964.3| 971.5| 1000.1| 967.9| 1047.3| 1087.0|
Table 9.

| Algorithm | N  | Median | Mean Rank | Z-Value |
|-----------|----|--------|-----------|---------|
| GA        | 10 | 1052.48| 29.5      | 2.81    |
| ICPSO     | 10 | 995.44 | 21.4      | 0.28    |
| MDPSO     | 10 | 999.44 | 19.6      | -0.28   |
| MICPSO    | 10 | 979.97 | 11.5      | -2.81   |
| Overall   | 40 |        | 20.5      |         |
Table 10.

Null hypothesis: All medians are equal  
Alternative hypothesis: At least one median is different

| DF | H-Value | P-Value |
|----|---------|---------|
| 3  | 11.97   | 0.007   |
Selected Shop Floor Number = sf

Determine PSO parameters \( N, \omega, c_1, r_1, c_2, r_2, k_{iter, sf} \)

Generate \( N \) particles for initial swarm

\( F_{gbest} = \inf \)

For \( p_{current} \) in initial swarm:

- Assign \( X \) vector probabilities to \( p_{current} \) randomly
- Assign \( V \) as zero-vector
- Make sample values from \( X \) vector

Calculate \( F_{p_{current}} \)

\( F_{p_{current}} = F_{p_{current}} \)

if \( F_{p_{current}} \leq F_{gbest} \):

\( S_{best} = F_{p_{current}} \)

\( P_{best} = p_{current} \)

Else:

\( F_{gbest} = F_{p_{current}} \)

end if

While \( generation < k_{iter, sf} \):

\( F_{total} = 0 \)

For \( p_{current} \) in new swarm:

\( generation += 1 \)

\( r_1, r_2 = \) random number

Determine new velocity vectors \( V \)

Determine new position vectors \( X \)

Weight all probabilities to make sum to 1

Generate sample values of \( X \) with probabilities of position vector.

Run simulation and calculate new fitness for \( p_{current} \)

if \( F_{p_{current}} \leq F_{p_{best}} \):

\( F_{p_{best}} = F_{p_{current}} \)

\( P_{best} = p_{current} \)

else:

\( F_{best} = F_{p_{best}} \)

end if

if \( F_{p_{current}} \leq F_{g_{best}} \):

\( F_{g_{best}} = F_{p_{current}} \)

\( S_{best} = p_{current} \)

End if

if \( F_{p_{current}} \leq F_{p_{best}} \):

\( F_{p_{best}} = F_{p_{current}} \)

\( P_{best} = p_{current} \)

End if

\( F_{total} += F_{p_{current}} \)

end for

Algorithm 1.
Define PSO parameters (w, c1, c2, swarm_size)

\[ F_{\text{best}} = \inf \]

Generate initial swarm \( s_0 = (p_1, p_2, \ldots, p_{\text{swarm.size}}) \)

For \( P_{\text{da}}, P_{\text{dp}}, P_{\text{route}} \) generate random probabilities:

Define \( X_{\text{drule}}, X_{\text{droute}}, X_{\text{route}} \) values according to the probabilities

for \( p_{\text{current}} \) in swarm:

Evaluate fitness values

Save \( p_{\text{current}} \) as the \( p_{\text{best}} \) of that particle

Save velocity vector as 0

if \( F_{\text{p.current}} < g_{\text{best}} \):

\[ g_{\text{best}} = F_{\text{p.current}} \]

\[ p_{\text{best}} = p_{\text{current}} \]

while \( i < \text{iter.size} \):

\[ s = s - 1 \]

for \( p_{\text{current}} \) in \( s \):

\( r_1, r_2 \) random numbers

Update velocity vector

for each gene in particle:

\( r_3 \) random number

if \( r_3 < 0.25 \):

Update \( p_{\text{current}} \) value by using probs values

for each gene in \( p_{\text{current}} \):

\( r_4 \) random number

if \( r_4 < 0.25 \):

Update \( p_{\text{current}} \) value with associated \( p_{\text{best}} \) value

for each gene in \( p_{\text{current}} \):

\( r_5 \) random number

if \( r_5 < 0.25 \):

Update \( p_{\text{current}} \)’s value with associated \( g_{\text{best}} \) value

Evaluate fitness value for \( p_{\text{current}} \)

If \( F_{\text{p.current}} \leq F_{\text{p.best}} \):

\[ F_{\text{p.best}} = F_{\text{p.current}} \]

\[ p_{\text{best}} = p_{\text{current}} \]

else:

\[ p_{\text{best}} = p_{\text{best}} \]

End if

if \( F_{\text{p.current}} \leq F_{\text{g.best}} \):

\[ F_{\text{g.best}} = F_{\text{p.current}} \]

\[ g_{\text{best}} = p_{\text{current}} \]

End if

\[ F_{\text{total}} + = F_{\text{p.current}} \]

\[ i + = 1 \]

end for

Algorithm 2.
Define PSO parameters \( \{ w, c_1, c_2, \text{swarm\_size}\} \)

\( F_{\text{best}} = \text{inf} \)

Give higher probabilities for the expected better rules

Generate initial swarm \( s_0 = \{p_1, p_2, \ldots, p_{\text{swarm\_size}}\} \)

Define \( X_{\text{drule}}, X_{\text{dregule}}, X_{\text{route}} \) values according to the probabilities

for \( p_{\text{current}} \) in swarm:

- Calculate \( p_{\text{current}} \) fitness value
- Save \( p_{\text{current}} \) fitness as the \( p_{\text{best}} \) of that particle
  
  if \( F_{p_{\text{current}}} \leq F_{p_{\text{best}}} \):
    
    \( g_{\text{best}} = p_{\text{current}} \)
    
    \( p_{\text{best}} = p_{\text{current}} \)

while \( (i < \text{iter\_size}) \):

- \( s = s_i - 1 \)

  for \( p_{\text{current}} \) in \( s \):

    - Select \%25 of the genes for the inertia part of PSO
    - Replace selected genes with other possible values
      according to the probabilities
    - Select \%25 of the genes for the cognitive part of PSO
    - Replace selected genes with related \( p_{\text{best}} \) values
    - Select \%25 of the genes for the social part of PSO
    - Replace selected genes with related \( g_{\text{best}} \) values

    if \( F_{p_{\text{current}}} \leq F_{p_{\text{best}}} \):
      
      \( F_{p_{\text{best}}} = F_{p_{\text{current}}} \)
      
      \( p_{\text{best}} = p_{\text{current}} \)

    else:
      
      \( p_{\text{best}} = p_{\text{best}} \)

    End if

    if \( F_{p_{\text{current}}} \leq F_{g_{\text{best}}} \):
      
      \( F_{g_{\text{best}}} = F_{p_{\text{current}}} \)
      
      \( g_{\text{best}} = p_{\text{current}} \)

    End if

    \( F_{\text{total}} = F_{p_{\text{current}}} \)

    \( i = i + 1 \)

end for

Algorithm 3.
Biographies

Caner Erden, currently working as Assistant Professor in the Faculty of Applied Sciences, Sakarya University of Applied Sciences, Sakarya, Turkey. He worked as a research assistant of industrial Engineering at Sakarya University and researcher at Sakarya University Artificial Intelligence Systems Application and Research between 2012-2020. He holds a PhD degree in Industrial Engineering from Natural Science Institute Industrial Engineering Department, Sakarya University, Turkey with the thesis titled "Dynamic Integrated Process Planning, Scheduling and Due Date Assignment". His research interests include scheduling, discrete event simulation, meta-heuristic algorithms, modelling and optimization, decision-making under uncertainty, machine learning and resource allocation and rough sets.

Halil İbrahim Demir was born in Sivas, Turkey in 1971. In 1988 he received a full scholarship and entered Bilkent University, Ankara, Turkey, to study in the Industrial Engineering Department. He got his Bachelor of Science degree in Industrial Engineering in 1993. In 1994, he received a full scholarship for graduate study in the USA from the Ministry of Education of Turkey. In 1997 he received a Master of Science degree in Industrial Engineering from Lehigh University, Bethlehem, Pennsylvania, USA. He was then accepted to Northeastern University, Boston, Massachusetts for Ph.D. study. He finished his Ph.D. courses at Northeastern and completed a Ph.D. thesis at Sakarya University, Turkey in 2005 for a Ph.D. in Industrial Engineering. He obtained an academic position at Sakarya University as an Assistant Professor. His Research Areas are Production Planning, Scheduling, Application of OR, Simulation, Artificial intelligence techniques, Genetic algorithms, Artificial Neural Networks, Fuzzy Logic and Decision making.
Onur Canpolat is currently a Research Assistant in the Department of Industrial Engineering at Sakarya University, Turkey. He received BSc. and MSc. degree in Industrial Engineering from Sakarya University, Sakarya, Turkey in 2012 and 2016, respectively. He is currently a PhD student at the same university. His areas of interest include Multi-Criteria Decision Making, Operations Research, Fuzzy Logic, Process Planning, Scheduling and Optimization.