RAIRO-Oper. Res. 55 (2021) 61–81
https://doi.org/10.1051/ro/2020105

RAIRO Operations Research
www.rairo-ro.org

MODELING AND OPTIMIZATION OF BATCH PRODUCTION BASED ON LAYOUT AND CUTTING PROBLEMS UNDER UNCERTAINTY

Mohammadhossein Saeedi and Ramyar Feizi*

Abstract. This paper presents modeling and optimization of batch production based on layout, cutting and project scheduling problems by considering scenario planning. In order to solve the model, a novel genetic algorithm with an improvement procedure based on variable neighborhood search (VNS) is presented. Initially, the model is solved in small sizes using Lingo software and the combined (proposed) genetic algorithm; then the results are compared. Afterwards, the model is solved in large sizes by utilizing the proposed algorithm and simple genetic algorithm. The main findings of this paper show: (1) The suggested algorithm is valid and able to achieve optimal and near-optimal solutions. This conclusion was made after proving the validity of the proposed method by solving a case study by employing the classical method (employing Lingo 11). And when the results were compared with the ones obtained by the proposed algorithm, they were found to be the same in both cases. (2) The combined genetic algorithm is more effective in obtaining optimal boundaries and the solutions close to them in all cases compared to the classical (simple) genetic algorithm. In other words, the main finding of this paper is a combined genetic algorithm to optimize batch production modeling problems, which is more efficient than the methods provided in the literature.

Mathematics Subject Classification. 90B30, 90-08.

Received October 11, 2019. Accepted September 20, 2020.

1. Introduction

In recent years, many papers have been devoted to production planning and scheduling in various types of production systems. One type of these systems is batch production which is usually adopted in process industries. This system has been at the center of attention of many researchers and is very significant in industrial production [4].

Production planning and scheduling is an activity that aims to use machine productions to the highest degree in order to meet production objectives during a certain time period called planning horizon [2]. Many researchers have contributed to developing the associated scheduling methods and terms.

Dai *et al.* [3] worked on stabilizing batch-processing networks. They showed that any normal dispatch policy (such as static buffer priority (SBP) and first-in–first-out (FIFO)) can be changed into a batch policy that keeps key stability properties. Traumann and Schwindt [12] presented a priority-rule based method in process

Keywords. Genetic algorithms, project scheduling, batch production, scenarios, layout and cutting problems.

Department of Industrial, Manufacturing & Systems Engineering, Texas Tech University, Box 43061, Lubbock, TX 79409-3061, USA.
*Corresponding author: ramyar.feizi@gmail.com

industries for batch production scheduling. They focused on a problem aiming to minimize the makespan in scheduling a given group of operations in a multipurpose batch plant. Liao and Liao [7] considered the problem of scheduling jobs in a flow shop with two batch processing machines in order to minimize the makespan. To this end, they presented Improved Mixed Integer Linear Programming (MILP) models for the problem. Muthuswamy *et al.* [9] examined the problem of minimizing makespan in a two-machine no-wait flow shop with two batch processing machines. Lu *et al.* [8] discussed the challenges brought by lead time on a Rolling Horizon Basis in a research which was conducted as a case study on the issues happened in a multi-product multi-stage manufacturing system. To address this problem, they considered it as a MILP problem and then modeled the problem which sought to find optimal production lots [8]. Hill *et al.* [5] investigated efficient batch scheduling in a petrochemical blending plant and proposed an effective scheduling heuristic for realistic production planning.

Also, some other researchers have modeled batch production problem in the form of resource-constrained project scheduling. Behnamian *et al.* [2] attempted to use mathematical model frameworks offered for multi-mode resource-constrained project scheduling so as to model batch production scheduling problem. This approach provided a more realistic model to the problem by considering factors such as transportation capacity and time. Sabzehparvar and Seyed-Hosseini [10] mapped project scheduling problem onto layout and cutting problems. In fact, they considered the problem of multi-mode resource constrained project scheduling. The objective of this problem was to minimize the project duration by determining a start time and a mode for each activity.

Table 1 was developed to provide a better understanding of the literature review and make a comparison among the previous studies. As it is shown in this table, the application of genetic algorithm in addressing batch production problems with the combination of neighborhood search operators is a new procedure. In addition, there is no research considering modeling of batch production under uncertainty as it can be understood from the table.

Therefore, to address the current gap in the literature, the current paper has designed a novel model for batch production problems under uncertainty in process industries. The main contribution of this paper is a mathematical model developed for the aforementioned problem as well as a novel genetic algorithm to solve the model. This proposed genetic algorithm is combined with neighborhood search operators in the form of variable neighborhood search. Afterwards, in order to evaluate the efficiency of the proposed genetic algorithm, the results of this algorithm are compared with those of simple genetic algorithm.

2. Problem description and mathematical modeling

The problem underlying this paper deals with modeling batch production under uncertainty. In this research, each batch consists of a number of products whose processing needs machines and time. The machines considered in this work are two types and renewable. It should be noted that the uncertainty in this study is considered discretely and based on different scenarios. In order to model the batch production problem, a combination of resource-constrained project scheduling problem and layout and cutting problems was used. For mathematical modeling, the batches were mapped onto activities, maximum time completion was mapped onto project execution duration, processing units and labor were mapped onto renewable resources. Moreover, each activity (batch) was assumed to be a box in three-dimensional space whose dimensions are time, resource #1 and resource #2. The purpose of the proposed model is to schedule the batches production/to determine the activities sequence/to determine the boxes layout so that the maximum completion time/the project execution duration/the sum of the widths of the laid-out boxes is minimized. The similarities in and mapping of the information of the problem in question based on layout and cutting problems as well as resource-constrained project scheduling in several modes are shown in Table 2.

2.1. Mathematical modeling

In this section, first, the indices, parameters and variables of the mathematical model are provided. Then the mathematical model is presented in the form of formulating objective function and the assumptions above are formulated as the model constraints.

TABLE 1. Comparison of the previous works in the literature.

Research (Alphabetic order)	Type of problem	Solution algorithm
Alizadeh and Husseinzadeh Kashan [1]	A batch processing machine scheduling with job conflicts and non-identical job sizes	Solved by two metaheuristic algorithms, namely the League Championship Algorithm and Optic Inspired Optimization
Behnamian *et al.* [2]	Single-batch scheduling by considering transportation among machines (factors such as transportation capacity and time)	Modeled by mixed integer programming Solved by considering lower bounds and heuristic algorithms
Damodaran [4]	Minimizing the makespan of parallel non-identical batch processing machines	Solved by a particle swarm optimization algorithm
Hill *et al.* [5]	Multi-product multi-BOM batch scheduling	The presented algorithm was based on a dynamic prioritization-based greedy search that scheduled the orders sequentially
Li and Zhang [6]	Single-batch processing machine scheduling with two-dimensional bin packing constraints	Modeled by a mixed integer programming Solved by heuristic algorithms, including four single-sequence based heuristics, a biased random-key genetic algorithm, and a hybrid bin loading algorithm
Liao and Liao [7]	Scheduling jobs in a flow shop with two batch processing machines	Formulated as Improved Mixed Integer Linear (MILP) Programming models The solution was proposed as a MILP-based heuristic algorithm
Lu *et al.* [8]	Scheduling of a multiproduct multi-stage batch processing system	Modeled as a mixed integer linear programming (MILP) problem The model was divided into several sub-problems as the horizon was rolled forward, of which a fix-and-relax strategy was applied
Muthuswamy *et al.* [9]	Makespan minimization in a two-machine no-wait flow shop with batch processing machines	A particle swarm optimization algorithm was proposed
Sabzehparvar and Seyed-Hosseini [10]	Multi-mode resource constrained project scheduling problem with mode dependent time lags	Solved by Floyd–Warshall algorithm
Tang and Liu [11]	Two-machine flow shop scheduling involving a batching machine with transportation or deterioration consideration	Solved by a heuristic algorithm and its worst-case performance was discussed
Traumann and Schwindt [12]	Scheduling a given set of operations in a multipurpose batch plant	Modeled by a novel two-phase approach (algorithm) dealing with two types of constraints separately
H. Zhou *et al.* [13]	Batch-processing machine scheduling with arbitrary release times and non-identical job sizes	A particle swarm optimization algorithm was modified and applied
S. Zhou *et al.* [14]	Parallel batch processing machine scheduling considering electricity consumption cost	A multi-objective differential evolution algorithm was proposed

Table 2. Mapping batch production, project scheduling, layout and cutting problems.

Batch production problem	Project scheduling problem	Layout and cutting problems
Batch or operations	Activities	Box
Maximum completion time	Project duration	Box width
Labor	Renewable resources	Box length
Down time	Activity calendar	
Identical processing units	Renewable resources	Box height
Machinery cleaning time	Setup time	
Multi-processing system	Correspondent/Similar machines	
Alternative processing units	Execution modes	Layout
Batches processing schedule	Activities sequence	Boxes

2.2. Indices and parameters

n: Number of batches (activities).
i: Index of batch or activity.
M: Number of alternative processing units (execution modes).
S: Number of scenarios.
s: Scenario index.
R_{1s}: Available amount of time of machine #1 in scenario s.
R_{2s}: Available amount of time of machine #2 in scenario s.
d^s_{imi}: Amount of time needed to process the activity (batch) in execution mode m_i (processing unit m_i) in scenario s.
r^s_{i1mi}: Amount of time needed from machine #1 as a resource to process the activity (batch) in execution mode m_i (processing unit m_i) in scenario s.
r^s_{i2mi}: Amount of time needed from machine #2 as a resource to process the activity (batch) in execution mode m_i (processing unit m_i) in scenario s.

2.3. Variables

x^s_{imi}: Start time of processing the activity (batch) in execution mode m_i (processing unit m_i) in scenario s.
y^s_i: Time allocated from machine #1 to process the activity (batch) in scenario s.
z^s_i: Time allocated from machine #2 to process the activity (batch) in scenario s.
t^s_{xij}: This is a binary variable. In scenario s, the activity (batch) i is equal to 0 if it is placed on the left of the activity (batch) j, otherwise it is equal to 1 (in fact, if the activity i is the predecessor of the activity j or it is processed before the activity j).
t^s_{xji}: This is a binary variable. In scenario s, the activity (batch) j is equal to 0 if it is placed on the left of the activity (batch) i, otherwise it is equal to 1 (in fact, if the activity j is the predecessor of the activity i or it is processed before the activity i).
t^s_{yij}: This is a binary variable. In scenario s, the activity (batch) i is equal to 0 if it is placed on the left of the activity (batch) j and uses machine #1 before the activity j, otherwise it is equal to 1 (in fact, if the activity i is the predecessor of the activity j or it is processed before the activity j).
t^s_{yji}: This is a binary variable. In scenario s, the activity (batch) j is equal to 0 if it is placed on the left of the activity (batch) i and uses machine #1 before the activity j, otherwise it is equal to 1 (in fact, if the activity j is the predecessor of the activity i or it is processed before the activity i).
t^s_{zij}: This is a binary variable. In scenario s, the activity (batch) i is equal to 0 if it is placed on the left of the activity (batch) j and uses machine #2 before the activity j, otherwise it is equal to 1 (in fact, if the activity i is the predecessor of the activity j or it is processed before the activity j).

t^s_{zji}: This is a binary variable. In scenario s, the activity (batch) j is equal to 0 if it is placed on the left of the activity (batch) i and uses machine #2 before the activity i, otherwise it is equal to 1 (in fact, if the activity j is the predecessor of the activity i or it is processed before the activity i).

V^s_{imi}: This is a binary variable and it is equal to 1 if the activity (batch) in execution mode m_i (processing unit m_i) is performed in scenario s, otherwise it is 0.

T_s: Duration of the project execution in scenario s.

W_s: This variable is binary, which is equal to 1 if the project is executed in scenario s, otherwise it is equal to 0.

2.4. Main structure of the mathematical model

Given the indices, parameters and variables, the mathematical model is as follows:

$$\min z = \sum_{s=1}^{S} W_s T_s. \tag{2.1}$$

The equation above represents the objective function which minimizes the project execution time or processing all the batches of the products.

Constraints:

$$T_s - x^s_i - \sum_{mi=1}^{M} d^s_{imi} \times v^s_{imi} \geq 0. \tag{2.2}$$

Relationship (2.2) calculates the project completion (execution) time which is equal to the maximum completion time. It also ensures that the execution time of each activity in any chosen scenario is not more than the duration of the project execution.

$$R_{1s} - y^s_i - \sum_{mi=1}^{M} r^s_{i1mi} \times v^s_{imi} \geq 0. \tag{2.3}$$

Relationship (2.3) ensures that the amount of time allocated from machine #1 in any mode and scenario to each activity is not greater than the amount available in each scenario.

$$R_{2s} - z^s_i - \sum_{mi=1}^{M} r^s_{i2mi} \times v^s_{imi} \geq 0. \tag{2.4}$$

Relationship (2.4) ensures that the amount of time allocated from machine #2 in any mode and scenario to each activity is not greater than the amount available in each scenario.

$$x^s_j - x^s_i - \sum_{mi=1}^{M} d^s_{imi} \times v^s_{imi} + \text{big}M \times t^s_{xij} \geq 0 \tag{2.5}$$

$$x^s_i - x^s_j - \sum_{mj=1}^{M} d^s_{jmj} \times v^s_{jmj} + \text{big}M \times t^s_{xji} \geq 0. \tag{2.6}$$

Relationships (2.5) and (2.6) ensure that processing the batches or activities (the boxes layout) doesn't overlap in terms of execution time (the box width layout) and priorities are met according to the schedule and input information (precedence relationships).

$$y^s_j - y^s_i - \sum_{mi=1}^{M} r^s_{i1mi} \times v^s_{imi} + \text{big}M \times t^s_{yij} \geq 0 \tag{2.7}$$

$$y^s_i - y^s_j - \sum_{mj=1}^{M} r^s_{j1mj} \times v^s_{jmj} + \text{big}M \times t^s_{yji} \geq 0. \tag{2.8}$$

Relationships (2.7) and (2.8) ensure that processing the batches or activities (the boxes layout) doesn't overlap in terms of assigning machine #1 (the box length layout) and priorities are met according to the schedule and input information (precedence relationships).

$$z_j^s - z_i^s - \sum_{mi=1}^{M} r_{i2mi}^s \times v_{imi}^s + \text{big}M \times t_{zij}^s \geq 0 \tag{2.9}$$

$$z_i^s - z_j^s - \sum_{mj=1}^{M} r_{j2mj}^s \times v_{jmj}^s + \text{big}M \times t_{zji}^s \geq 0. \tag{2.10}$$

Relationships (2.9) and (2.10) ensure that processing the batches or activities (the boxes layout) doesn't overlap in terms of assigning machine #2 (the box length layout) and priorities are met according to the schedule and input information (precedence relationships).

$$\sum_{mi=1}^{M} v_{imi}^s = 1. \tag{2.11}$$

Relationship (2.11) ensures that each activity (batch) is performed in only one mode.

$$x_i^s \leq \text{big}M \times w_s \tag{2.12}$$

$$y_i^s \leq \text{big}M \times w_s \tag{2.13}$$

$$z_i^s \leq \text{big}M \times w_s \tag{2.14}$$

$$v_{imi}^s \geq w_s \tag{2.15}$$

$$T_s \leq \text{big}M \times w_s. \tag{2.16}$$

Relationships (2.12)–(2.16) ensure that the project is executed in only one scenario.

$$x_i^s, y_i^s, z_i^s, T_s \geq 0 \tag{2.17}$$

$$w_s, t_{xij}^s, t_{xji}^s, t_{yij}^s, t_{yji}^s, t_{zij}^s, t_{zji}^s, v_{imi}^s = \{0, 1\}. \tag{2.18}$$

Relationships (2.17) and (2.18) specify the allowable ranges of values for the decision variables.

3. Solution algorithm

This section proposes a combined genetic structure to optimize the objective function considered in the model. The purpose of this algorithm is to better achieve global solutions which is a combination of genetic algorithm and an improvement procedure.

3.1. Proposed structure

The general structure of the proposed genetic algorithm is as follows (Fig. 1):

The pseudocode above shows the proposed structure of the combined genetic algorithm.

In this research, a matrix structure is used to display the solution and an example of such matrix is shown in Figure 2. This figure is a two-dimensional matrix whose first row shows the activities sequence and its second row shows the execution mode corresponding to each activity.

J_i is the activity (batch) number that is scheduled at the ith location of the sequence.

It should be noted that the way the scheduling is performed is according to the model constraints as well as the allocation of the machines and the absence of the interference of the x, y and z dimensions.

```
Genetic algorithm
{Initialization:
 Initialize algorithm parameters
Generate N feasible solutions as initial population
While the stopping criterion is met do
Calculate solution fitness for each one of the N solutions
Calculate the number of the solutions (nc) for crossover operator
Counter =0
While counter<=nc do
Select two parents
Counter=counter+2
Apply crossover operator to the selected parents
End while
Calculate the number of solutions (nm) for mutation operator
Counter=0
While counter<=nm do
Select one solution that was not selected
Apply mutation operator to the selected solution
Counter=counter+1
End while
Apply reproduction operator to the other solutions in the population that were not selected
Apply improved method
Select the best N solutions as the population of the next generation
End while
Return the best solutions}
```

FIGURE 1. Pseudocode of the combined genetic algorithm.

J_1	J_2	J_3	.....	J_i	.....	J_n
M_1	M_2			M_i		M_n

FIGURE 2. The way of displaying the solution.

3.2. Generation of initial solutions

Parallel neighborhood search method was used to generate initial solutions. This method includes two neighborhood search operators which act in parallel (simultaneously). The procedure of the parallel neighborhood search is that a feasible solution is first generated randomly and then this generated solution is inserted as an input into the parallel neighborhood search method. Finally, the output solution of the parallel neighborhood search method is added to the set of the generated solutions if it is not repetitive. The procedure of the neighborhood search operators is as follows:

The first neighborhood search operator: In this operator, two indices such as i and j are generated randomly in the uniform interval $[1, n]$ (n is the number of the batches produced) and columns i and j are replaced with each other. Then given the new order of the batches, the start time of processing each batch as well as the variables y and z are determined.

The second neighborhood search operator: In this operator, two indices such as i and j are generated randomly in the uniform interval $[1, n]$ (n is the number of the batches produced) and the columns between i and j are reversed. Then given the new order of the batches, the start time of processing each batch as well as the variables y and z are determined.

The parallel neighborhood search method is a result of combining these two operators in parallel which is as follows:

1. Start.
2. Set counter to 0.
3. Generate a random feasible solution such as s.
4. Insert input solution s into the first operator and name output solution s_1.
5. Insert input solution s into the second operator and name output solution s_2.
6. Given the minimum value of the objective function, select one of the solutions s, s_1, s_2 and name it s.
7. Add one unit to the counter.
8. If the counter is more than the upper limit of the number of iterations, go to 8 and otherwise go to 3.
9. Report the solution s as the output of the procedure.
10. End.

The parallel neighborhood search method is performed $2N$ times and then N solutions are selected as the initial population among the non-repetitive generated solutions. To choose the solutions, the $2N$ solutions obtained by the parallel neighborhood search method are put in a set and sorted in ascending order of the objective function value. Afterwards, the first N solutions which have lower values are chosen as the set of the initial solutions.

3.3. Intersect operator

In this work, a single-point intersect operator was used. First, an index i is generated randomly in the interval $[1, n]$. Then columns 1 to i for each row of the first child inherit the values of columns 1 to i from the first parent and take the rest of the sequence from the second parent. In other words, the activities which are not in the first parent are added to the first child in the same sequence as they belong to the second parent. The second child is formed in the same shape as the first child except that the places of the first parent and second parent are switched. The procedure of the intersect operator is shown in Table 3. With regard to Table 3, suppose the problem has six activities and three execution modes. Also, assume the index i is equal to 2.

3.4. Jump operator

In order to implement jump operator, a parallel neighborhood search structure was created in which three neighborhood search operators were used in parallel. The neighborhood search operator consisting of the two operators explained in the previous section is as follows:

Third neighborhood search operator: This operator generates an index sen in the uniform interval $[1, S]$ (S represents the number of scenarios). If the previous scenario (the scenario of project execution which is in

TABLE 3. Procedure of the intersect operator.

The first parent					
1	2	4	3	5	6
1	3	2	1	3	1
The second parent					
1	3	2	4	5	6
1	2	3	3	2	1
The first child					
1	2	3	4	5	6
1	3	3	3	2	1
The second child					
1	3	4	3	5	6
1	2	2	1	3	1

the input solution) is different from sen, it is replaced with sen. Then given the new scenario, the start time of processing the batches (x) and variables (y, z) is calculated.

3.5. Improvement procedure

An improvement procedure was created which was applied to the solutions and improved them as much as possible. This procedure is based on variable neighborhood search (VNS). Also, two neighborhood search structures (NSS) were created and combined with VNS structure. The neighborhood search structures used are the same as the three neighborhood search operators described in the previous section.

As it was stated, two neighborhood search structures were combined with variable neighborhood search (VNS) structure. This combination is as follows:

```
{for each input solution S
K = 1
While the stopping criterion is met do
    S1 = Apply mutation type K
    S = Acceptance method (S, S1)
    If S is improved then
      K = 1
      Else
        K = k + 1
        If K = 4 then
          K = 1
        Endif
End while}
```

As it can be seen in the structure above, after applying the neighborhood structure to the solution, the acceptance procedure was applied to the solution obtained and the previous solution. Then one of these two solutions was chosen as the solution of the next iteration of VNS. The acceptance procedure acts according to the objective function value, that is, the solution with the minimum objective function value is accepted.

3.6. Choosing next-generation solutions

As it was explained, genetic algorithm works with a population of solutions whose size is constant in all iterations or generations. At the end of each iteration, it is necessary that the population of the solutions of the next iteration be determined. In addition, all the existing solutions (the current population and newly generated

Table 4. Execution time of the activities/batches in each mode and scenario.

Activity	Mode	Scenario 1	Scenario 2
Activity 1	Mode 1	2	2
	Mode 2	3	3
Activity 2	Mode 1	4	5
	Mode 2	2	3
Activity 3	Mode 1	5	4
	Mode 2	3	2
Activity 4	Mode 1	3	4
	Mode 2	4	4

Table 5. The amounts of time of machines needed in each mode and scenario.

		Machine 1		Machine 2	
Activity	Mode	Scenario 1	Scenario 2	Scenario 1	Scenario 2
Activity 1	Mode 1	2	3	3	5
	Mode 2	3	2	4	2
Activity 2	Mode 1	2	1	3	1
	Mode 2	2	2	3	2
Activity 3	Mode 1	4	5	3	4
	Mode 2	3	2	2	5
Activity 4	Mode 1	1	4	2	1
	Mode 2	2	4	3	3

solutions) are put in a solution pool and then are sorted in ascending order according to the objective function value. After sorting the solutions, the first N (the size of the population) solutions are selected as the solutions of the population of the next generation or iteration.

4. Computational results

The proposed algorithm and simple genetic algorithm were implemented in MATLAB 2016. Several sample problems were created to solve the model using the algorithm and after adjusting the parameters of the model and algorithm, these problems were solved by the proposed algorithm and simple genetic algorithm. Adjusting the parameters and characteristics of the sample problems will be explained in the following sections. Then the results of solving these problems and the results of solving a small-sized problem using Lingo will be examined.

4.1. The results of solving the problems using Lingo

In order to examine and evaluate the validity of the proposed mathematical model, it was coded using Lingo 11. Moreover, the model was run and solved by considering a small-sized problem which is explained in the next paragraph.

The problem consists of four activities, two execution modes, two machines and two scenarios. The available amounts of time of machine #1 in scenarios #1 and #2 are ten and fourteen respectively. Also, the available amounts of time of machine #2 in scenarios #1 and #2 are twelve and fifteen respectively (Tabs. 4, 5).

After solving the model for the example described above and determining the optimal solution, the characteristics of the obtained optimal solution were provided as shown in Table 6 and Figs. 3, 4.

TABLE 6. Obtained optimal solution using Lingo.

Activity	Scenario	Mode	x	y	z
1	1	1	0	2	3
2	1	2	8	0	0
3	1	2	2	7	6
4	1	1	5	4	10

FIGURE 3. Layout of the activities or batches with respect to x and y dimensions.

FIGURE 4. Layout of the activities or batches with respect to x and z dimensions.

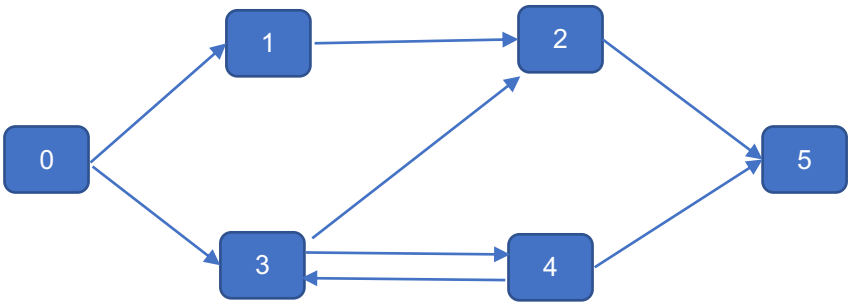

FIGURE 5. Graph of the relationships between the batches or activities.

TABLE 7. Execution time of the activities/batches in each mode and one scenario.

Activity	Mode	Scenario 1
Activity 1	Mode 1	2
	Mode 2	3
Activity 2	Mode 1	3
	Mode 2	1
Activity 3	Mode 1	1
	Mode 2	3
Activity 4	Mode 1	2
	Mode 2	4

TABLE 8. The amounts of time of machines needed in each mode and one scenario.

Activity	Mode	Machine 1	Machine 2
		Scenario 1	Scenario 1
Activity 1	Mode 1	2	2
	Mode 2	2	3
Activity 2	Mode 1	3	2
	Mode 2	2	1
Activity 3	Mode 1	3	2
	Mode 2	2	2
Activity 4	Mode 1	4	3
	Mode 2	2	1

It should be noted that the optimal value of the objective function is 10. This sample problem was also solved by the combined genetic algorithm whose results were consistent with the results of solving it using Lingo software.

In addition, the sample problem solved in the research by Sabzehparvar and Seyed-Hosseini [10] was solved using Lingo software and the combined genetic algorithm. This example consists of six batches that are numbered from 0 to 5. In this problem, batches #0 and #5 are dummy and in fact, it can be said that this problem has four batches or activities. Moreover, since different scenarios were not considered in the model presented in Sabzehparvar and Seyed-Hosseini's paper [10], it is assumed that there is only one scenario in solving this example (Fig. 5 and Tabs. 7, 8).

TABLE 9. Obtained optimal solution using Lingo.

Activity	Scenario	Mode	x	y	z
1	1	1	1	2	1
2	1	2	4	2	3
3	1	1	0	1	0
4	1	2	1	0	0

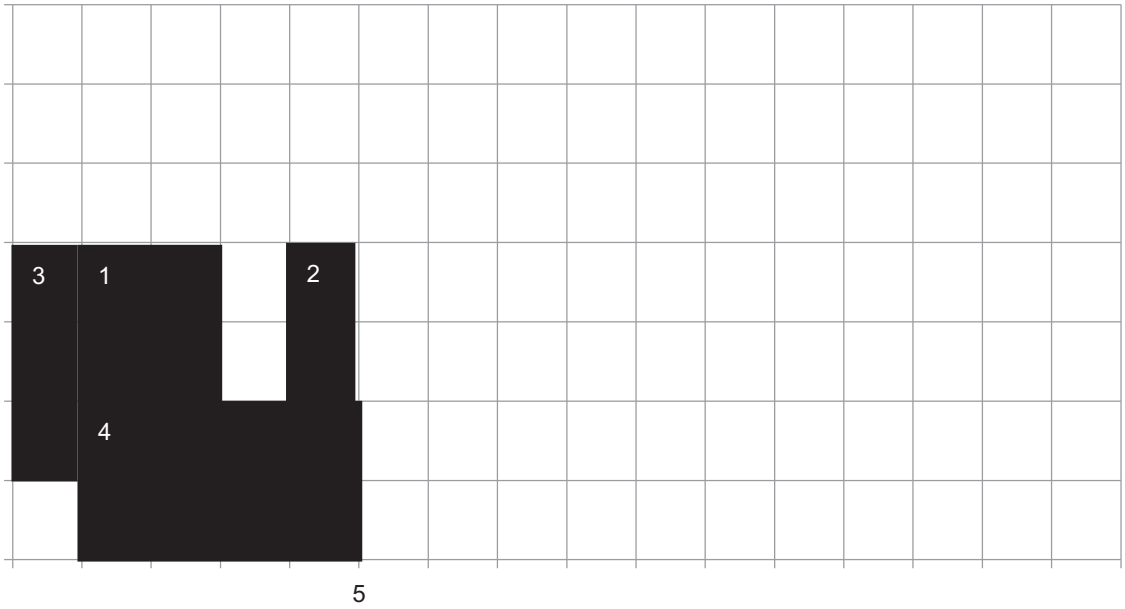

FIGURE 6. Layout of the activities or batches with respect to x and y dimensions.

After solving the model for the example described above and determining the optimal solution, the characteristics of the obtained optimal solution were provided as shown in Table 9 and Figs. 6, 7.

It should be noted that the optimal value of the objective function is 5. This sample problem was also solved using the combined genetic algorithm whose results were consistent with the results of solving it using Lingo software.

Furthermore, several problems were solved by Lingo software whose characteristics are explained separately as follows.

Problem #1: In this problem, there are six batches or activities (four main batches and two dummy batches), two scenarios and two execution modes. In addition, the available amount of time of the first machine is four and that of the second machine is three (Tabs. 10–12 and Figs. 8, 9).

It should be noted that the optimal value of the objective function is four. This sample problem was also solved by the combined genetic algorithm whose results were consistent with the results of solving it using Lingo software.

Problem #2: In this problem, there are six batches or activities (four main batches and two dummy batches), two scenarios and three execution modes. The available amount of time of the first machine is four and that of the second machine is three (Tabs. 13–15 and Figs. 10, 11).

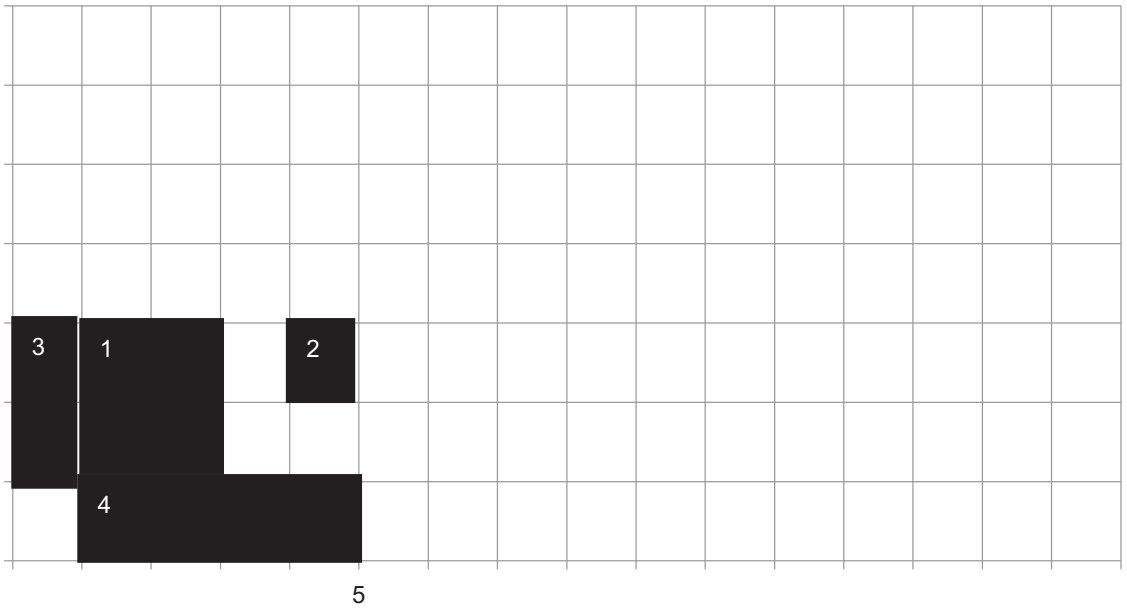

FIGURE 7. Layout of the activities or batches with respect to x and z dimensions.

TABLE 10. Execution time of the activities/batches in each mode and scenario.

Activity	Mode	Scenario 1	Scenario 2
Activity 1	Mode 1	1	2
	Mode 2	3	3
Activity 2	Mode 1	2	3
	Mode 2	1	2
Activity 3	Mode 1	1	2
	Mode 2	3	3
Activity 4	Mode 1	2	2
	Mode 2	3	4

TABLE 11. The amounts of time needed from the machines in each mode and scenario.

Activity	Mode	Machine 1		Machine 2	
		Scenario 1	Scenario 2	Scenario 1	Scenario 2
Activity 1	Mode 1	2	1	3	2
	Mode 2	2	1	2	1
Activity 2	Mode 1	2	2	2	2
	Mode 2	3	2	1	2
Activity 3	Mode 1	1	3	2	3
	Mode 2	2	2	2	2
Activity 4	Mode 1	4	2	1	1
	Mode 2	2	3	1	2

TABLE 12. Obtained optimal solution using Lingo.

Activity	Scenario	Mode	x	y	z
1	1	1	0	0	0
2	1	2	1	1	2
3	1	1	1	0	0
4	1	1	2	0	0

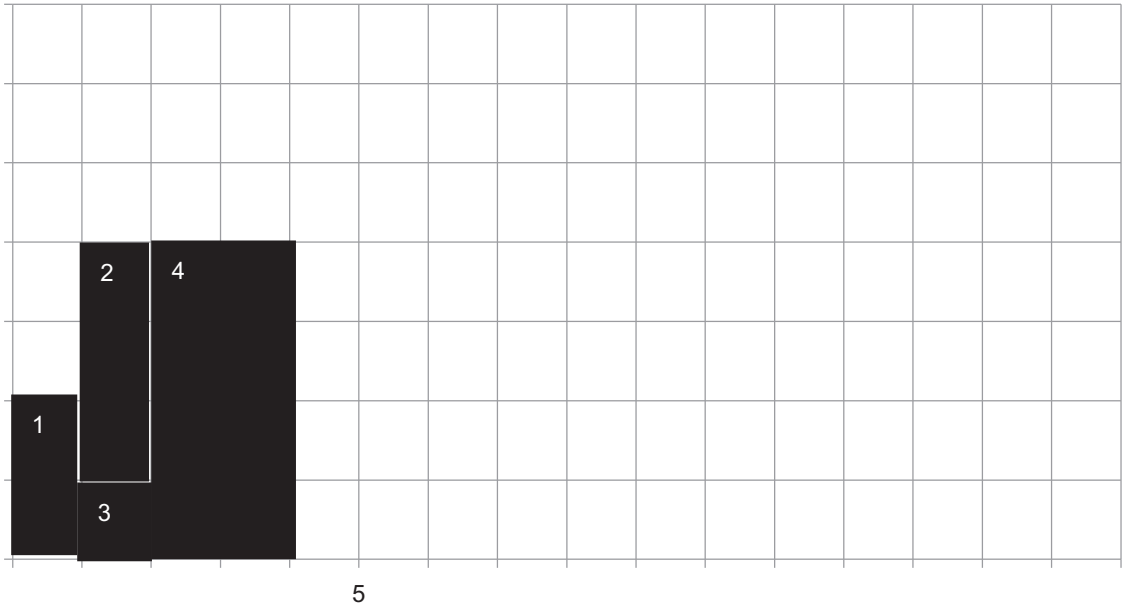

FIGURE 8. Layout of the activities or batches with respect to x and y dimensions.

The optimal value of the objective function is 4. This sample problem was also solved by the combined genetic algorithm whose results were consistent with the results of solving it using Lingo software.

In addition to the problems described above, another problem with twelve activities was chosen to solve using Lingo software which has one scenario, three resources and fourteen activities (two dummy activities and twelve main activities). The characteristics of the problem are described as follows (Fig. 12).

Problem #3: In this problem, there are three types of resources ($K = 3$) and there are five machines from the first resource, four machines from the second resource and four machines from the third resource. The number of the machines used to perform the activities in any time period should not exceed the total number of the existing machines. Also, each batch or activity needs one unit from each resource. The problem has two execution modes in which the duration of performing each activity in each execution mode is as follows (Tab. 16):

First, it was attempted to solve the problem using Lingo software. However, it was found that Lingo was not able to find the feasible solution after a long solving time and the problem was solved only by the combined genetic algorithm. The solution results are shown in Table 17.

It should be noted that the objective function value is 24.

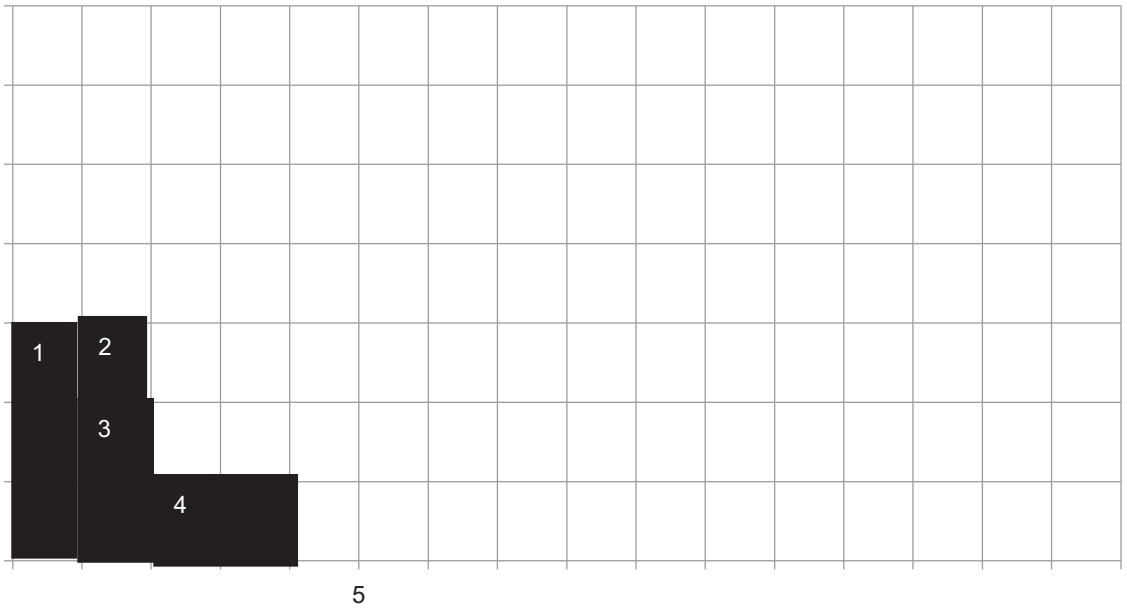

FIGURE 9. Layout of the activities or batches with respect to x and z dimensions.

TABLE 13. Execution time of the activities/batches in each mode and scenario.

Activity	Mode	Scenario 1	Scenario 2
Activity 1	Mode 1	1	1
	Mode 2	3	3
	Mode 3	2	1
Activity 2	Mode 1	1	3
	Mode 2	2	2
	Mode 3	2	1
Activity 3	Mode 1	1	2
	Mode 2	3	1
	Mode 3	2	1
Activity 4	Mode 1	2	2
	Mode 2	3	3
	Mode 3	2	2

4.2. Results evaluation of the solution algorithm

In this section, a number of problems of different sizes were created and solved by the combined and simple genetic algorithms. The characteristics of the problems are shown in Table 18.

Adjusting the model parameters to solve the sample problems is as follows:

- The amounts of time needed from the machines for processing are in the uniform interval [5, 10].
- The amounts of time of the available machines are in the interval $[0.75a, 1.5a]$ where a is the total needed from each machine in each scenario and mode.
- The time to perform the activities is generated in the uniform interval [1, 100].
- The population size is 200 and the number of the algorithm iterations is 500.

TABLE 14. The amounts of time needed from the machines in each mode and scenario.

Activity	Mode	Machine 1		Machine 2	
		Scenario 1	Scenario 2	Scenario 1	Scenario 2
Activity 1	Mode 1	2	1	3	2
	Mode 2	2	1	2	2
	Mode 3	2	2	2	1
Activity 2	Mode 2	2	2	2	2
	Mode 1	3	2	1	2
	Mode 3	2	1	2	2
Activity 3	Mode 1	1	3	2	3
	Mode 2	2	2	2	1
	Mode 3	2	2	2	2
Activity 4	Mode 1	4	4	1	3
	Mode 2	2	3	1	2
	Mode 3	1	2	3	3

TABLE 15. Obtained optimal solution using Lingo.

Activity	Scenario	Mode	x	y	z
1	2	3	0	0	0
2	2	3	1	0	0
3	2	2	1	1	2
4	2	3	2	0	0

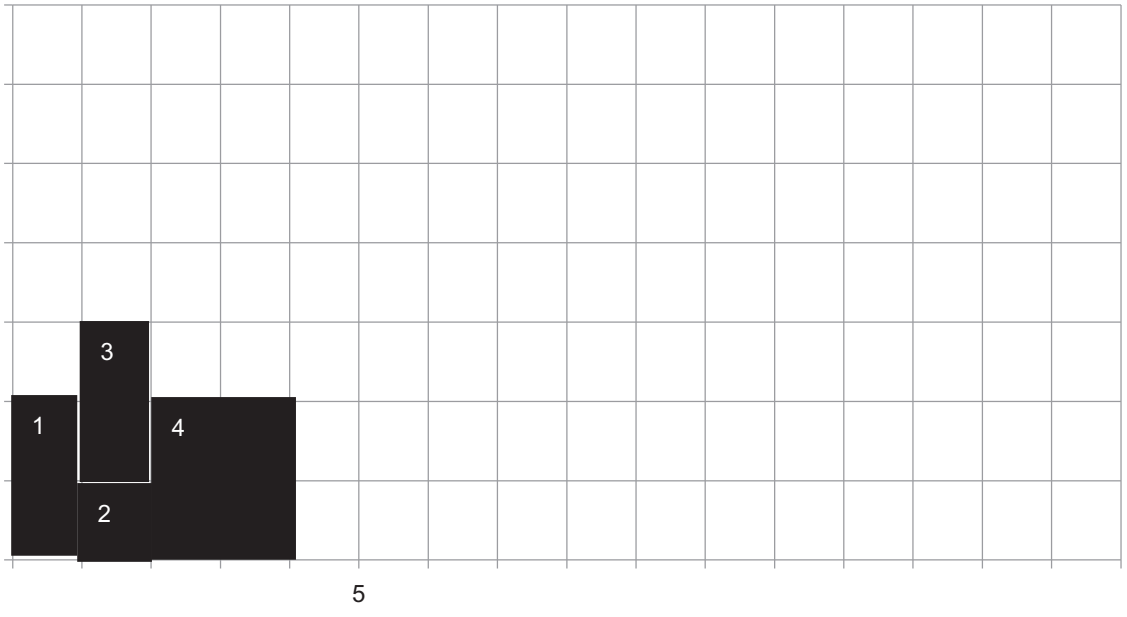

FIGURE 10. Layout of the activities or batches with respect to x and y dimensions.

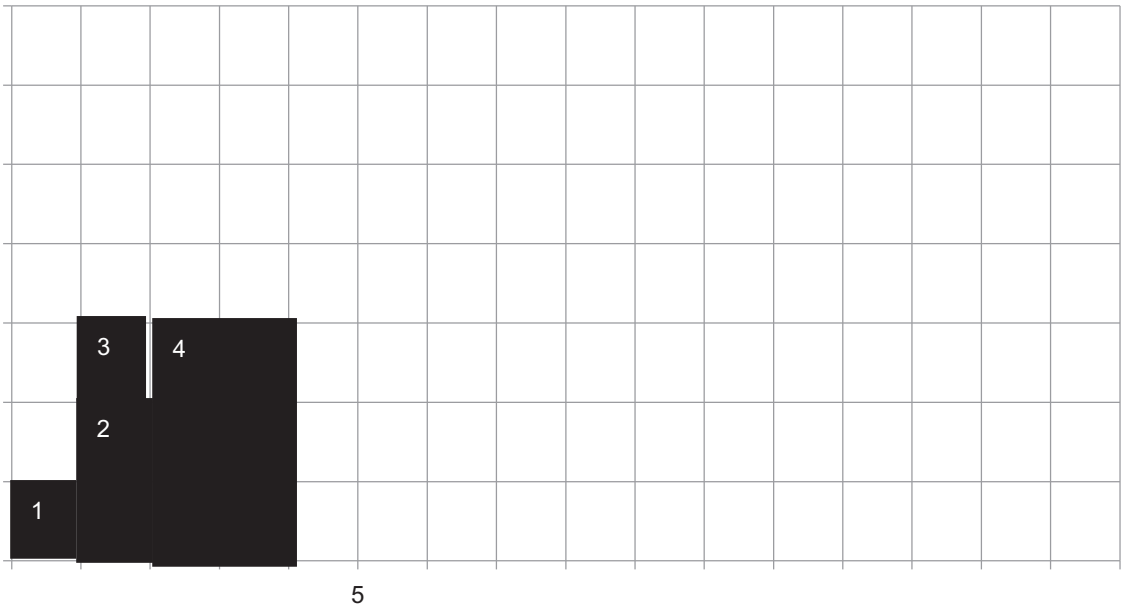

FIGURE 11. Layout of the activities or batches with respect to x and z dimensions.

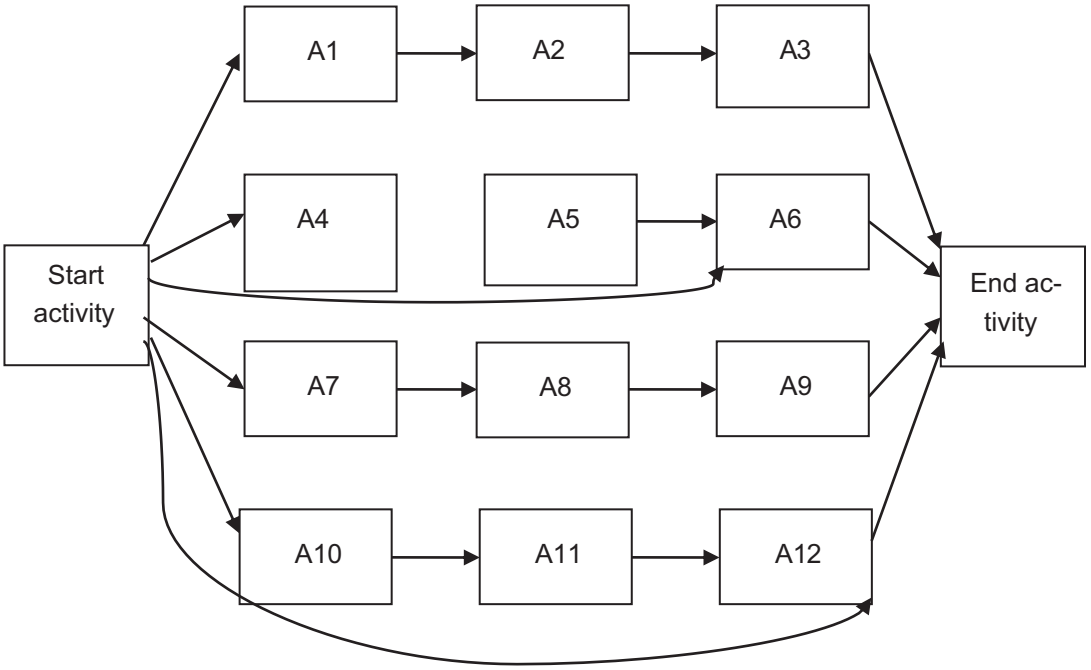

FIGURE 12. AoN network of the production project activities of the product.

TABLE 16. Execution time of the activities in the execution modes.

Activity	Mode	Execution time	Activity	Mode	Execution time (in seconds)
1	Mode 1	10	7	Mode 1	4
	Mode 2	7		Mode 2	3
2	Mode 1	9	8	Mode 1	12
	Mode 2	8		Mode 2	11
3	Mode 1	4	9	Mode 1	10
	Mode 2	4		Mode 2	5
4	Mode 1	10	10	Mode 1	12
	Mode 2	9		Mode 2	7
5	Mode 1	8	11	Mode 1	4
	Mode 2	5		Mode 2	2
6	Mode 1	12	12	Mode 1	15
	Mode 2	10		Mode 2	10

TABLE 17. Results of solving the problem using the combined genetic algorithm.

Activity	Scenario	Mode	x	y	z	w
1	1	2	0	1	1	1
2	1	2	7	1	1	1
3	1	2	15	1	1	1
4	1	2	0	0	0	0
5	1	2	9	0	0	0
6	1	2	14	0	0	0
7	1	2	0	3	3	3
8	1	2	3	3	3	3
9	1	2	14	3	3	3
10	1	2	0	2	2	2
11	1	2	7	2	2	2
12	1	2	13	2	2	2

TABLE 18. Sample problems.

The number of scenarios	The number of modes	The number of batches (activities)	Problem number
1	3	10	1
2	3	10	2
3	3	10	3
1	3	20	4
2	3	20	5
3	3	20	6
1	3	40	7
2	3	40	8
3	3	40	9
1	3	70	10
2	3	70	11
3	3	70	12
1	3	100	13
2	3	100	14
3	3	100	15

Table 19. Results of solving the sample problems.

Problem number	Combined genetic algorithm		Simple genetic algorithm	
	Objective function Value	Time (In seconds)	Objective function Value	Time (In seconds)
1	518	0.0988	543	0.0343
2	534	0.053	540	0.026
3	690	0.1001	745	0.0538
4	1115	0.289	1278	0.099
5	1143	0.299	1321	0.1
6	1017	0.298	1229	0.11
7	1679	0.66	2098	0.39
8	1670	0.66	2106	0.36
9	1665	0.72	2081	0.4
10	2251	1.004	2942	0.77
11	2243	0.98	2869	0.73
12	2199	1.009	2937	0.78
13	3815	1.73	4061	0.88
14	3822	2.17	4116	0.92
15	3884	2.44	4476	1.29

– The jump and intersection rates are set to be 0.1 and 0.8 respectively.

After solving the sample problems, the results were displayed as shown in Table 19.

According to the results in Table 19, it is observed that the objective function value obtained by the combined genetic algorithm is less than the corresponding value obtained by the simple genetic algorithm in all the problems. Based on this result, it can be said that the combined genetic algorithm is more effective in achieving solutions compared to the simple or classical genetic algorithm. In fact, the combined genetic algorithm with the proposed structure is more efficient than the classical or simple genetic algorithm. Also, comparing the implementation times reveals that the duration of solving the problems using the classical genetic algorithm in all cases is less than that of the combined genetic algorithm. The reason for this seems to be the time-consuming improvement procedure existing in the structure of the proposed algorithm.

5. Conclusion

In this paper, a novel model was proposed for batch production problems under uncertainty. Each batch includes several products that require machines and time to process. The machines considered in this study were two types and renewable. It should be noted that the uncertainty was considered discretely and based on different scenarios. To model the batch production, a combination of resource-constrained project scheduling, layout and cutting problems was used.

To solve the proposed model, a novel algorithm was proposed with a combined structure for its implementation. In this algorithm, parallel and variable neighborhood search operators were used. Furthermore, in order to achieve optimized solutions, the algorithm was combined with an improvement procedure based on variable neighborhood search procedure.

Afterwards, the validity and accuracy of the model and the combined genetic algorithm were evaluated. This was done by creating and solving a small-sized problem using Lingo 11 software and the proposed algorithm after modeling and implementing the algorithm in MATLAB. The results of solving the model using Lingo 11 are an indication of the feasibility of the obtained solution and the validity of the model. In addition, since the results of solving this sample problem using the proposed algorithm were identical to the results obtained by

Lingo 11, it can be said that the proposed algorithm is valid and able to accurately search the solution area and achieve optimal or near-optimal solution.

After investigating the validity of the proposed model and algorithm, several sample problems in different sizes were created and after adjusting the algorithm and model parameters, they were solved by the classical and combined genetic algorithms. Then the results of these algorithms were compared based on the objective function values and solving time. Comparing the results of the two algorithms revealed that the combined genetic algorithm performs better in achieving optimized solutions than the simple or the classical genetic algorithm. In fact, the combined genetic algorithm with the proposed structure is more efficient than the classical or simple genetic algorithm. Also, comparing implementation time showed that the solving time of the classical genetic algorithm in all cases was less than that of the combined genetic algorithm. This seems to be a result of the time-consuming process of the improvement procedure in the structure of the proposed algorithm.

Finally, future studies can be conducted to not only apply the proposed models and solution algorithms, but also to improve it in other ways. As the proposed method is applied in more cases, it can provide a better understanding of the managerial insights of the model. Furthermore, doing so can improve the method by lowering the computation time as the problem size increases.

References

[1] N. Alizadeh and A. Husseinzadeh Kashan, Enhanced grouping league championship and optics inspired optimization algorithms for scheduling a batch processing machine with job conflicts and non-identical job sizes. *Appl. Soft Comput. J.* **83** (2019).

[2] J. Behnamian, S.M. Fatemi Ghomi, F. Jolai and O. Amirtaheri, Realistic two-stage flowshop batch scheduling problems with transportation capacity and time. *Appl. Math. Modell.* **36** (2012) 723–735.

[3] J.G. Dai and C. Li, Stabilizing batch-processing networks. *Oper. Res.* **51** (2003) 123–136.

[4] P. Damodaran, D.A. Diyadawagamage, O. Ghrayeb and M.C. Velez-Galleg, A particle swarm optimization algorithm for minimizing makespan of non-identical batch processing machines in parallel. *Int. J. Adv. Manuf. Technol.* **58** (2012) 1131–1140.

[5] A. Hill, T. Cornelissens and K. Sörensen, Efficient multi-product multi-BOM batch scheduling for a petrochemical blending plant with a shared pipeline network. *Comput. Chem. Eng.* **84** (2016) 493–506.

[6] X. Li and K. Zhang, Single batch processing machine scheduling with two-dimensional bin packing constraints. *Int. J. Prod. Econ.* **196** (2018) 113–121.

[7] C.J. Liao and L.M. Liao, Improved MILP models for two-machine flowshop with batch processing machines. *Math. Comput. Modell.* **48** (2008) 1254–1264.

[8] S. Lu, H. Su, Y. Wang, L. Xie and Q. Zhang, Multi-product multi-stage production planning with lead time on a rolling horizon basis. In: *9th International Symposium on Advanced Control of Chemical Processes The International Federation of Automatic Control June 7–10. Whistler* (2015).

[9] S. Muthuswamy, J. Maya, M. Rojas-Santiago and C.M. Velez-Gallego, Minimizing makespan in a two-machine no-wait flow shop with batch processing machines. *Int. J. Adv. Manuf. Technol.* **63** (2012) 281–290.

[10] M. Sabzehparvar and S. Seyed-Hosseini, A mathematical model for the multi-mode resource-constrained project scheduling problem with mode dependent time lags. *J. Supercomput.* **44** (2008) 257–273.

[11] L. Tang and P. Liu, Two-machine flowshop scheduling problems involving a batching machine with transportation or deterioration consideration. *Appl. Math. Model.* **33** (2009) 1187–1199.

[12] N. Traumann and C. Schwindt, Priority-rule based scheduling of chemical batch processes. *Comput. Aided Chem. Eng.* **21** (2006) 2165–2170.

[13] H. Zhou, J. Pang, P.-K. Chen and F.-D. Chou, A modified particle swarm optimization algorithm for a batch-processing machine scheduline problem with arbitarary release times and non-identical job sizes. *Comput. Ind. Eng.* **123** (2018) 67–81.

[14] S. Zhou, X. Li, N. Du, Y. Pang and H. Chen, A multi-objective differential evolution algorithm for parallel batch processing machine scheduling considering electricity consumption cost. *Comput. Oper. Res.* **96** (2018) 55–68.