

A new method for mathematical and simulation modelling interactivity: A case study in flexible job shop scheduling

Ojstersek, R.^{a,*}, Lalic, D.^b, Buchmeister, B.^a

^aUniversity of Maribor, Faculty of Mechanical Engineering, Maribor, Slovenia

^bUniversity of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia

ABSTRACT

The present study has investigated mathematical and simulation model interactivity for production system scheduling. A mathematical model of a Flexible Job Shop Scheduling Production optimisation problem (FJSSP) was used to evaluate a new evolutionary computation method of multi-objective heuristic Kalman algorithm (MOHKA). Ten Brandimarte and five Kacem benchmarks were applied for evaluation and comparison of MOHKA optimisation results with the Multi-Objective Particle Swarm Optimization algorithm (MOPSO) and Bare-Bones Multi-Objective Particle Swarm Optimization algorithm (BBMOPSO). Benchmark data sets were divided into three groups, regarding their complexity, from low, middle to high dimensional optimisation problems. The optimisation results of MOHKA show high capability to solve complex multi-objective optimisation problems, especially with real world production systems data. A new robust method is presented of optimisation data interactivity between a mathematical optimisation algorithm and a simulation model. The results show that the presented method can overcome the integrated decision logic of commercial simulation software and transfer the optimisation results into the simulation model. Our interactive method can be used in a variety of production and service companies to ensure an optimised and sustainable cost-time profile.

© 2019 CPE, University of Maribor. All rights reserved.

ARTICLE INFO

Keywords:

Flexible job shop scheduling;
Mathematical modelling;
Simulation modelling;
Interactivity;
Evolutionary computation;
Multi-objective heuristic Kalman algorithm (MOHKA);
Multi-objective particle swarm optimization (MOPSO);
Bare-bones multi-objective particle swarm optimization algorithm (BBMOPSO)

*Corresponding author:
robert.ojstersek@um.si
(Ojstersek, R.)

Article history:

Received 15 May 2019
Revised 21 November 2019
Accepted 7 December 2019

1. Introduction

In the time of Industry 4.0 [1], which represents highly flexible manufacturing systems, proper scheduling of high-mix low-volume production systems orders is crucial [2]. The high degree of complexity and flexibility of multi-objective optimisation problems increases the problem to achieve optimal scheduling of such production systems significantly [3]. The use of conventional methods [4] and their obtained optimisation results does not achieve the optimally weighted goals of production systems related to appropriate makespan, uniformly high machinery utilisation, financially and timely justified production. For many years, scientists have been using evolutionary computation (EC) methods [5] for the purpose of multi-objective production systems' optimisation [6]. They have been struggling to transfer mathematical models of algorithms into simulation environments to achieve optimised real world production systems indirectly. The complexity of transferring mathematical models into the simulation environments is constrained by the incompatibility of programming environments, integrated decision models within simulation environments, and problems in the reliability of optimisation results' transfers

into the simulation environment, in a real world production system. Relevance use and satisfactory optimisation results of EC methods in solving FJSSP are transmitted via the newly proposed algorithms [7, 8] to the real world environment of Industry 4.0 production systems [9, 10]. The complexity of the FJSSP optimisation problem is reflected in the multi-objective nature of the optimisation problem [11]. However, for the transfer and use of new EC methods in a real world environment, the extensive evaluation of algorithms using benchmark and real production datasets is crucial. In this case, the use of simulation modelling is crucial, both in designing the experiment, and in evaluating the optimisation results [12]. The main research problem relates to the compatibility and interactivity between the mathematical model of the EC method optimisation results and the simulation model, which enables the evaluation and application of the EC method in a real world environment [13]. The interactivity between EC methods and the simulation model poses major challenges for researchers, who have not yet studied them thoroughly. In the presented research work, a new method is presented of transferring optimisation results between a mathematical model of MOHKA's EC method [14] and a simulation model in a conventional Simio simulation environment. The presented method provides interactivity between the EC optimisation algorithm and the simulation model, while offering high flexibility of the simulation model and the integration of the EC algorithm optimisation results into the integrated decision logic of the simulation environment [15].

The manuscript is divided into the following sections. Section 2 deals with the mathematical definition of a multi-objective FJSSP optimisation problem. Section 3 presents our own developed method of the multi-objective EC method, called MOHKA. Examples are given of mathematical modelling and experimental testing on five Kacem [16] and ten Brandimarte [17] datasets. Analysis of the MOHKA algorithm optimisation results is performed in comparison with the two existing algorithms MOPSO [18] and BBMOPSO [19]. In this section also the interactivity of the MOHKA mathematical modelling optimisation results and simulation modelling is given. The new proposed block structure of an interactive simulation model method and integrating optimisation results into a simulation model is presented, based on the Kacem and Brandimarte input datasets. Graphically and numerically, the high ability is confirmed of interactivity with optimisation results and the simulation model. Section 3 overviews the new interactive method of using the MOHKA optimisation results in a simulation model, that allows direct use in a real world production system. The high ability to use the proposed method enables further research work on optimally implemented collaborative work places in flexible manufacturing systems.

2. Problem description

Production scheduling is defined as decision-making processes that are used on a daily basis in many production and service enterprises [20]. The importance of the decisions taken is, consequently, reflected in the fields of jobs orders, production, transport and distribution of the final products [21]. Production scheduling is the process of optimising, controlling and determination of the limited production system resources (machines, humans, finances etc.). The presented mathematical and simulation modelling method is based on solving an FJSSP multi-objective optimization problem. Given the high degree of difficulty, the following Section presents a notation of abbreviations, a general mathematical description of a multi-objective FJSSP optimization problem, and a mathematical approach how to solve it.

2.1 Notation

The notations presented by Graham *et al.* [22] will be used in the presented paper.

i	Job ($i = 1, \dots, n$)	j	Machine ($j = 1, \dots, m$)
k	Operation ($k = 1, \dots, O_i$)	h	Resource ($h = 1, \dots, s$)
n	Total number of jobs	m	Total number of machines
O_i	Number of operations of job J_i	s	Number of limited resources
p_i, p_{ij}	Processing time of job J_i on machine M_j	p_{ik}, p_{ikj}	Processing time of operation O_{ik} on M_j

f_1	Makespan (time required to complete all jobs)	f_2	Maximum workload (workload of the most loaded machine)
f_3	Total workload of all machines		

2.2 Multi-objective flexible job shop scheduling problem

Multi-objective FJSSP is described as: We have n jobs which can be performed on m machines from a set of machines ($j = 1, \dots, m$) suitable for carrying out the jobs. The choice of using the machine is made according to the machine occupancy and the suitability of the individual machines to perform the operation. The number of jobs n and number of machines m are given. Each job i has a specific sequence and number of operations O_i . The processing time of the operation p_{jk} may vary, depending on the machine on which it is performed. For the multi-objective FJSSP some limitations must be made:

- One machine can process only one job at a time.
- One job can be processed only on one machine at a time.
- When the operation starts it cannot be interrupted until the end of the operation, after the completion the next operation can start.
- All the jobs and operations have equal priorities at the time zero.
- Each machine m is ready at time zero.
- Given an operation O_{ij} and the selected machine m , the processing time p_{ij} is fixed.

Eqs. 1, 2, and 3 describe three optimisation objectives, as follows:

- Makespan (time required to complete all jobs):

$$f_1 = \max \{C_j \mid j = 1, \dots, n\} \tag{1}$$

- Maximum workload (workload of the most loaded machine):

$$f_2 = \max \sum_{i=1}^n \sum_{j=1}^{n_i} p_{ijk} x_{ijk}, k = 1, 2, \dots, m \tag{2}$$

- Total workload of all machines:

$$f_3 = \sum_{i=1}^n \sum_{j=1}^{n_i} \sum_{k=1}^m p_{ijk} x_{ijk}, k = 1, 2, \dots, m \tag{3}$$

where C_j is the completion time of job J_i , and x_{ijk} is a decision variable on which individual machine operation will be processed. In Table 1, we see the FJSSP benchmark example, presented by Kacem *et al.* [16]. This example has three jobs that must be processed on four machines. The processing time of each operation depends on the machine on which the operation will take place.

Fig. 1 shows the optimal solution to the FJSSP optimisation problem presented in Table 1.

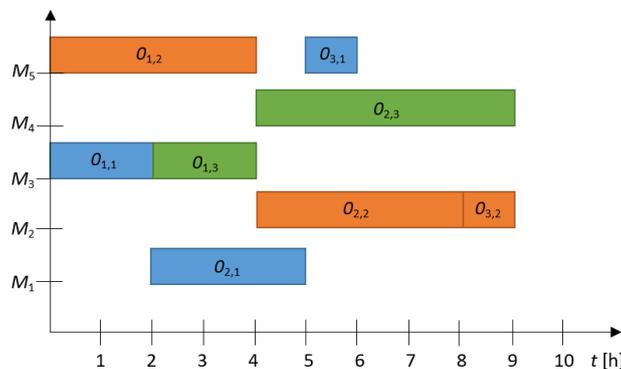


Fig. 1 Gantt chart of optimal solution for a FJSSP optimisation problem

Table 1 FJSSP benchmark example

	$O_{i,j}$	M_1	M_2	M_3	M_4	M_5
J_1	$O_{1,1}$	5	4	2	6	7
	$O_{2,1}$	3	5	7	8	4
	$O_{3,1}$	5	4	3	2	1
J_2	$O_{1,2}$	9	7	6	7	4
	$O_{2,2}$	5	4	7	8	5
	$O_{3,2}$	9	1	2	2	4
J_3	$O_{1,3}$	4	7	2	4	8
	$O_{2,3}$	5	5	7	5	7

From the optimisation problem defined above, the following limitations can be described:

- All machines are available at time $t = 0$, and any order J_j can be started at time $t = r_j$.
- Only one operation at a time can be performed by a machine. It becomes available to other operations only when the current operation on the machine is completed.
- Each operation $O_{i,j}$ is the start time $r_{i,j}$ defined as shown by Eqs. 4 and 5:

$$r_{1,j} = r_j, \forall 1 \leq j \leq n \text{ in } r_{i+1,j} = r_{i,j} + \gamma_{i,j}, \text{ where } \gamma_{i,j} = \min_k(d_{i,j,k}) \tag{4}$$

$$\forall 1 \leq i \leq n_j - 1, \forall 1 \leq j \leq n \tag{5}$$

2.3 Multi-objective optimisation

Multi-objective optimisation is an area that deals with multi-objective decision-making of mathematical optimisation problems [23]. Optimisation problems involve more than one optimisation function, where several variables of the optimisation problem need to be optimised. A feature of multi-objective optimisation is that there is not only one optimal solution optimising the optimisation function [24], but, for these functions, there are infinitely many Pareto optimal solutions [25]. Pareto solutions are non-dominant, Pareto optimal or Pareto effective [26]. All Pareto optimal solutions in the Pareto space are considered equally appropriate. The field of Multi-objective optimisation is increasingly present in everyday life. Multi-objective optimisation can be found in all fields of Sciences, Economics, Logistics, and where it is necessary to make optimal decisions in the presence of trade-offs between two or more conflicting goals [27].

In a mathematical sense, a multi-objective problem can be formulated with Eq. 6:

$$\min(f_1(x), f_2(x), \dots, f_k(x)); x \in X, \tag{6}$$

where the integer $k \geq 2$ represents the number of optimisation parameters, and X represents the feasible set of decision vectors. A set of decision vectors is usually represented by constraint functions. We define the vector-valued objective function as shown in Eq. 7:

$$f: X \rightarrow \mathbb{R}^k, f(x) = (f_1(x), \dots, f_k(x))^T. \tag{7}$$

If we want to maximise a function, we can do it by minimising its negative dependence. The element $x^* \in X$ presents a workable solution or a workable decision. The vector $z^* = f(x^*) \in \mathbb{R}^k$ is called the function vector for the feasible solution x^* . In multi-objective optimisation, there is usually no viable solution that optimises all of the target functions at the same time. Pareto optimal solutions are solutions that cannot be improved without compromising at least one of the goals of the remaining functions.

Using the mathematical notations of Eqs. 8 and 9, we can conclude that the feasible solution $x^1 \in X$ Pareto is dominated by another solution $x^2 \in X$ in the case where:

$$f_i(x^1) \leq f_i(x^2) \text{ for all } i \in \{1, 2, \dots, k\} \text{ and} \tag{8}$$

$$f_i(x^1) < f_i(x^2) \text{ for at least one } j \in \{1, 2, \dots, k\} \tag{9}$$

is a feasible solution, $x^* \in X$ and the associated output value $f(x^*)$ is Pareto optimal if there is no other solution that dominates it. The Pareto group of optimal solutions is called the Pareto Front. The Pareto Front of multi-objective optimisation problems is limited by two vectors:

- The nadir vector is defined by Eq. 10:

$$z_i^{nad} = \sup_{x \in X} f_i(x) \text{ for all } i = 1, \dots, k \tag{10}$$

- The ideal vector is defined by Eq. 11:

$$z_i^{ideal} = \inf_{x \in X} f_i(x) \text{ for all } i = 1, \dots, k \tag{11}$$

The nadir and ideal vector components define the upper and lower bounds for the optimisation functions of Pareto optimal solutions [28].

3. Results and discussion

3.1 Multi-objective heuristic Kalman algorithm (MOHKA)

In order to solve the planning and scheduling problem of FJSSP, researchers use different evolutionary computation methods for solving the multi-objective nature of the problem. The presented research work is based on the use of the Multi-Objective Heuristic Kalman Algorithm (MOHKA), which is based on the mathematical formulation of the Heuristic Kalman Algorithm (HKA), whose operation and use is presented in the cited literature [29]. The positive results of the algorithm's operation and the developed model architecture make it possible to upgrade single-objective optimisation to multi-objective optimisation. MOHKA, the same as HKA, first in the basic interaction, evaluates the solutions using a Gaussian distribution based on the parameters given in the variance and convergence matrix. Based on the definition of the multi-objective Pareto optimal solutions, a limiting function is determined, to ensure that the solutions obtained are within the appropriate limits. The nadir vector z_i^{ideal} and the ideal vector z_i^{ideal} , are determined defined by Eqs. 10 and 11. After the evaluation of the suitability of the obtained solutions is completed, the solutions are written to a non-dominant function, which is stored in the data set of non-dominant solutions. The evaluation procedure follows, and the maximum number is selected of the discussed optimisation problem articles . In the final step, an evaluation of the obtained results is performed using a random distribution function. The pseudocode of the algorithm, handling the constraints, updating the non-dominated solution archive, choosing the best samples and pruning the non-dominated solution archive are presented in the literature [14]. Fig. 2 presents a general flow chart of a MOHKA optimisation algorithm.

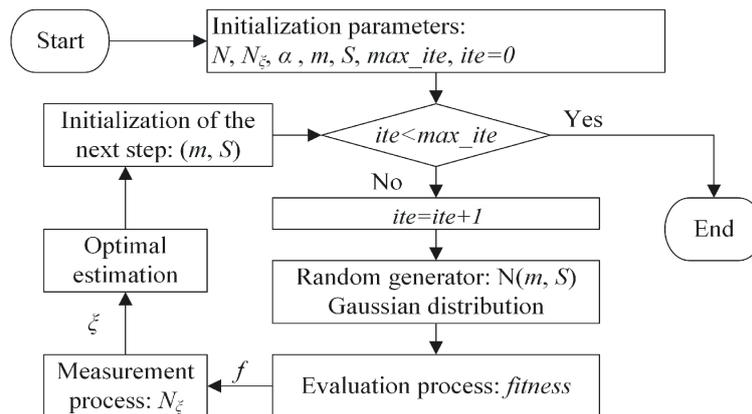


Fig. 2 Flow chart of MOHKA optimisation algorithm

Mathematical modelling of experiments

To test the performance of the MOHKA algorithm, we used two of the most well established benchmark data sets for multi-objective optimization of FJSSP. We used five Kacem datasets (Kacem 4×5, Kacem 8×8, Kacem 10×7, Kacem 10×10 and Kacem 15×10) [16] and ten Brandimarte datasets (Mk01 to Mk10) [17]. We divided these benchmark data sets into three groups, according to the complexity of the optimisation problem. Considering the recommenda-

tions in the literature [30], we divided the datasets into low, middle and high dimensional optimisation problems. The division of benchmark datasets according to the complexity of the optimisation problems allows a more accurate evaluation of the obtained results, in order to determine the advantages and limitations of the mathematically modelled optimisation algorithm. With the MOHKA algorithm, we optimised three key parameters of a flexible production system: Makespan (MC), total workload of all machines (TW) and maximum workload of an individual machine (MW). The obtained results of the algorithm were compared with the optimal results of the FJSSP problem obtained in the literature [31]. The algorithm was implemented in the MATLAB R2017b software environment, using a PC with an Intel i7 processor with 16 GB of working memory.

Mathematical modelling results

In order to compare the obtained MOHKA solutions, in addition to the Pareto optimal results, we selected the two currently most advanced algorithms for solving FJSSP optimisation problems: Multi-Objective Particle Swarm Optimization Algorithm (MOPSO) [18] and an improved Bare-Bones Multi-Objective Particle Swarm Optimization (BBMOPSO) algorithm [19]. According to the recommendations [14], the initial parameters of the MOHKA algorithm were set to: $N = 300$, $N_{\xi} = 10$, $\alpha = 0.3$, $Na = 100$, $m_r = 0.1$, and $MaxIter = 3000$. The MOPSO and BBMOPSO parameter settings are the same as those in the literature [18, 19].

The optimisation results presented in Tables 3, 4, and 5 represent the optimal solutions of three algorithms, MOHKA, MOPSO, and BBMOPSO, compared to the Pareto optimal solutions. The graphical symbols of the optimisation algorithms on the graphs are presented in Table 2.

In order to demonstrate their high ability to solve low, middle and high complex optimisation problems, the following results are presented separately. Numerical and graphical results are given, and comparative comments, advantages and limitations of individual algorithms' comparisons are described. Numerical values MC , TW , MW , presented in Tables 3, 4, 5, are average values of all obtained optimisation results.

Table 2 Graphic symbols of the optimisation algorithms

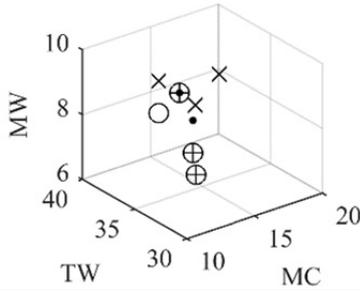
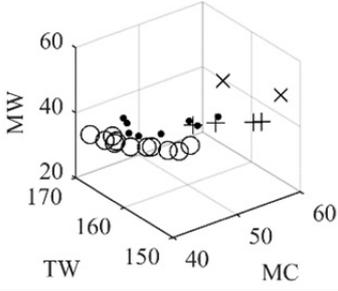
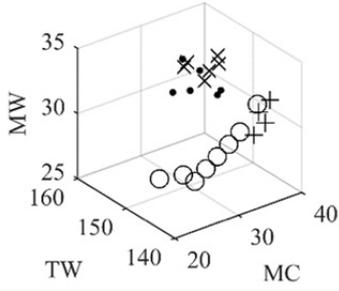
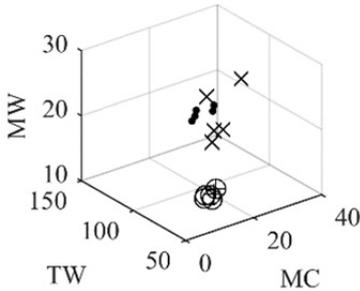
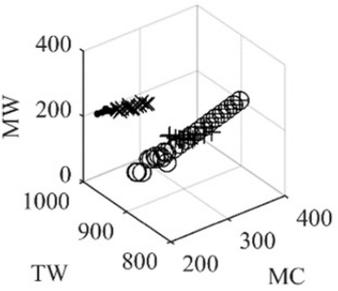
Symbol	•	x	+	o
Algorithm	MOHKA	MOPSO	BBMOPSO	Pareto optimum

- Low dimensional optimisation problems

We notice that, with Kacem benchmark datasets, MOHKA solves low-dimensional optimisation problems better. In Table 3, MOHKA achieves Pareto an optimal solution for the Kacem 4×5 dataset, where the MOHKA solutions are located directly in the centre of the Pareto optimal solutions. Compared to MOPSO and BBMOPSO, the Mk01 dataset is also solved better, where the distance between MOHKA optimisation solutions and Pareto optimal solutions are shorter than with the other two algorithms. With the Mk02 dataset, the Pareto optimisation solutions are the closest to the results of the BBMOPSO optimisation algorithm. The comparison between MOHKA and MOPSO shows slightly more optimal solutions given by the MOHKA algorithm. The Kacem 8×8 dataset again demonstrates the equivalence of optimisation results between the MOHKA and MOPSO algorithms. BBMOPSO solved this test data set with an optimally defined solution. BBMOPSO near-optimal optimization results were also presented in the Mk03 dataset, where its solutions are on the Pareto front of optimal solutions. With regard to the MOHKA and MOPSO algorithm solutions, we can claim a similar ability to solve the given data set.

In general, we find that the MOHKA algorithm in these low dimensional problems solves the optimization problems as satisfactorily and comparatively as the mentioned comparative algorithm. In two cases, the Kacem 4×5 and Mk01, MOHKA was the most successful, in the others it was comparable to MOPSO and slightly worse than BBMOPSO.

Table 3 Optimisation results of low dimensional problems

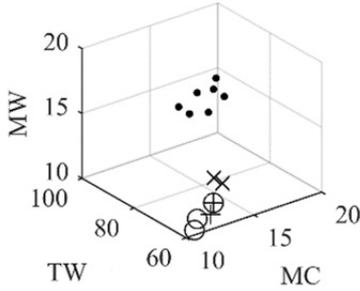
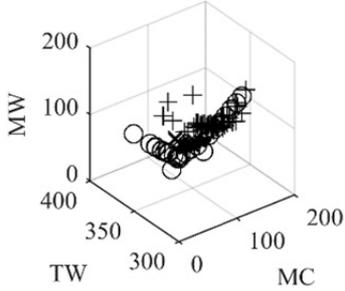
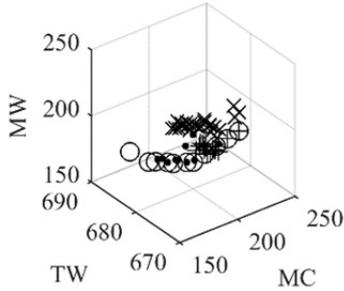
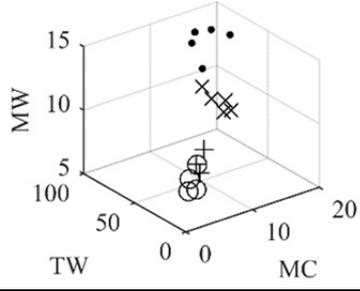
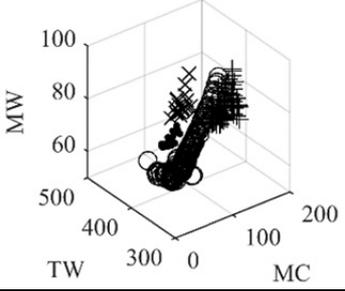
	Kacem			Brandimarte					
									
	(a) Kacem 4×5			(b) Mk01			(c) Mk02		
	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>
•	11.5	32	9.5	47.1	161.9	38.4	36.8	157.3	30
+	12	32.3	8.3	52	153.8	43.3	34.5	141	31.5
x	14.7	35.7	9	57	156	48.5	38.3	158	30.8
									
	(d) Kacem 8×8			(e) Mk03					
	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>			
•	25	115.8	19.2	224.3	979.7	218.7			
+	16	74	12.5	273	809.2	254.1			
x	25.6	101.2	20	263.6	973.1	209.1			

• Middle dimensional optimisation problems

In the optimisation results of the middle dimensional optimisation problems, presented in Table 4, we find that MOHKA had the largest deviation from the Pareto optimal solutions for the two Kacem datasets in the Pareto graph of solutions Kacem 10×7 and Kacem 10×10. The graphical and numerical results of MOPSO and BBMOPSO proved more optimal optimisation results compared to MOHKA. With the Kacem 10×7 dataset, BBMOPSO achieved the Pareto optimal solution and MOPSO came very close to it. Similarly, the optimisation algorithms solved the Kacem 10×10 dataset, where the difference between MOHKA and MOPSO was smaller, which proves the relevance and comparability of MOHKA with MOPSO. The MOHKA optimisation algorithm solved the Brandimarte middle dimensional optimisation problems better. We can see that, for the two Mk04 and Mk05 datasets, MOHKA solved both data sets most satisfactorily, since the MOHKA algorithm solutions are on the Pareto front of optimal solutions. For the two datasets listed above, BBMOPSO had the most problems, presented on the Mk04 dataset, in which solutions were the furthest from the Pareto optimal solutions. In the case of the Mk05 dataset, MOHKA was comparable to MOPSO. The Mk06 data set was solved relatively well by the MOHKA and BBMOPSO algorithms, with MOPSO farthest from the optimal solution.

We concluded that MOHKA performs best in the middle dimensional optimisation algorithms in the Brandimarte datasets, where it dominates the solutions of the other two algorithms. The Kacem datasets were the furthest from the optimal solutions, compared to BBMOPSO and MOPSO.

Table 4 Optimisation results of middle dimensional problems

	Kacem			Brandimarte					
									
	(f) Kacem 10×7			(g) Mk04			(h) Mk05		
	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>
•	16.6	87.4	15.3	85.9	343.4	74.9	189.5	679.3	181.5
+	12	60.5	11.5	118.1	342.4	100.5	201.5	676.3	189.9
x	13.5	66	12.5	93.3	350.1	70.8	213.5	681.9	189.5
									
	(i) Kacem 10×10			(j) Mk06					
	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>			
•	16.6	88.4	13	104.5	444.9	61.7			
+	8.7	42.7	7	123.6	357.1	84.8			
x	14.6	63	9.8	123.6	446.8	72.9			

• High dimensional optimisation problems

MOHKA optimisation algorithm problems in solving middle dimensional Kacem optimisation problems continued with high dimensional optimisation problems, in Table 5, which can be attributed to the construction of Kacem datasets, that are highly susceptible to dropping optimisation algorithms to local minima, to which MOHKA is highly exposed by the KA mathematical structure [14]. Due to the hybridisation and mathematical structure of the algorithm, Kacem 15×10 was best solved by BBMOPSO. We see that all three algorithms were at a relatively short distance from the Pareto optimal solution. Unlike Kacem datasets, MOHKA excelled at Brandimarte datasets, which is more important for solving FJSSP optimisation problems. The structure of the Brandimarte datasets represents real world dataset input of an FJSSP production system. In our case, where we wanted to solve the real world problems of scheduling manufactured systems and establish communication between the optimisation algorithm and the simulation model, this is crucial [32]. The success of solving Brandimarte datasets is paramount in solving FJSSP. We see that MOHKA solved Brandimarte high dimensional optimisation problems Mk08 and Mk09 perfectly for data sets Mk09, and especially Mk08, where we see that MOHKA generated an optimal solution with a high degree of solution delivery reproducibility. Such results demonstrate the robustness and high ability to solve the FJSSP optimisation problem.

From presented solutions, we find that MOHKA optimisation results are comparable to the results of MOPSO and BBMOPSO algorithms, especially in Brandimarte datasets, where MOHKA was the most suitable algorithm for solving both medium and high dimensional optimisation problems. MOHKA also demonstrated its competitiveness in low dimensional Kacem datasets. More limitations and deviations from optimal solutions can be detected in medium and high dimensional Kacem datasets.

Table 5 Optimisation results of high dimensional problems

Kacem			Brandimarte						
(l) Kacem 15×10			(m) Mk07			(n) Mk08			
	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>
•	24.8	171	21.3	178.2	689.9	169.8	550.1	2506.5	547.5
+	16	91.5	14	189.3	666.2	167.3	597.4	2504	551.9
x	23.4	135.3	19	207	693.7	174.7	599.3	2523.5	545.1
(o) Mk09			(p) Mk10						
	<i>MC</i>	<i>TW</i>	<i>MW</i>	<i>MC</i>	<i>TW</i>	<i>MW</i>			
•	350.5	2450.9	306.5	358.1	2074.5	226.8			
+	534.9	2224.2	383.7	464.2	1888.9	298.8			
x	536.1	2448.2	361.2	429	2110.3	258.9			

3.2 Simulation modelling

In the time of the Industry 4.0, development of simulation tools to optimise production has become increasingly more important. Using advanced simulation methods, we can make manufacturing systems more efficient, financially viable and more competitive in the global market [33]. There are many optimisation problems in production systems that can be solved by the proper use of simulation tools [34]. In our case, we wanted to establish interactivity between the EC optimisation method and the flexible simulation model. The interactive architecture enables communication between the optimisation algorithm and the production system simulation model. Fig. 3 presents the proposed block architecture of a simulation model that enables the interactivity of MOHKA optimisation algorithm results, a simulation model and a real world production system.

The interactive architectural model consists of the following phases:

- In the first phase, we assign the input data of a real production system or benchmark data sets. The inputs of a real production system must be credible and verifiable.
- In the second phase, the implementation of the sequence order optimisation is performed according to the available machines with the MOHKA optimisation algorithm. The algorithm is implemented according to the structure described in Section three.
- The third phase transmits the MOHKA optimisation results to the simulation model of a real production system, named the analysed system. Additional optimisation parameters are assigned related to costs, dimensions and setup time. At this stage, we propose the introduction of a new approach for determining the order of operations on an individual machine, which does not depend on the integrated simulation environment decision logic. The approach is presented as follows.

- The next phase is the implementation of simulation experiments that allow the calculation of the average multiple iterations' values of the simulation model, and the introduction of simulation scenarios, in order to determine the limitations, changes and proposals for optimisation of the production system.
- In the simulation analysis phase, we evaluate the numerical results, and compare the graphical matching of order sequences solutions using MOHKA and the simulation model. Approval of the orders' sequences is confirmed or denied. The importance of data matching ensures the credibility of numerical and simulation results.
- The evaluated numerical and simulation results are transferred to a real production system, whereby a change (optimisation) of the production system is enabled, based on an interactive loop.

The simulation model was built in the software environment Simio [35], which is a unique software environment for modelling flexible manufacturing or service processes [36]. Simio is based on the use of intelligent objects, and supports both process and object oriented modelling. It is used for discrete and continuous systems. Using the MOHKA algorithm, we solved the FJSSP optimization problem, so we decided to upgrade our existing optimisation results with a suitable simulation model. The following is a new decision logic method that combines the possibilities of testing datasets and optimising real world production systems. The obtained numerical values of the optimisation algorithm, which are given in Table 6, were transferred into the Simio software environment, where the real production system shown in Fig. 4 was modelled.

The main advantage of using simulation environments, such as Simio, is the ability to transfer data between the optimisation algorithm, simulation model and real world production system. With the appropriate data transfer method, we can extend the testing and optimisation of production system parameters, in order to extend the numerical model in an optimisation-programming environment that optimises different orders. The simulation model was designed as a flexible modular model built in two parts. The first part of the simulation model is a MOHKA optimisation algorithm that allocates work orders optimally to available machines. In this case, MOHKA optimises three key parameters: Makespan, maximum workload and total workload of all machines. The end result of the MOHKA optimisation algorithm is the optimal allocation of individual work order operations to the available machines. In this case, the order machine execution sequence, their start time, finish time, and machine sequence are obtained. The output results of the MOHKA optimisation algorithm are shown in Table 6.

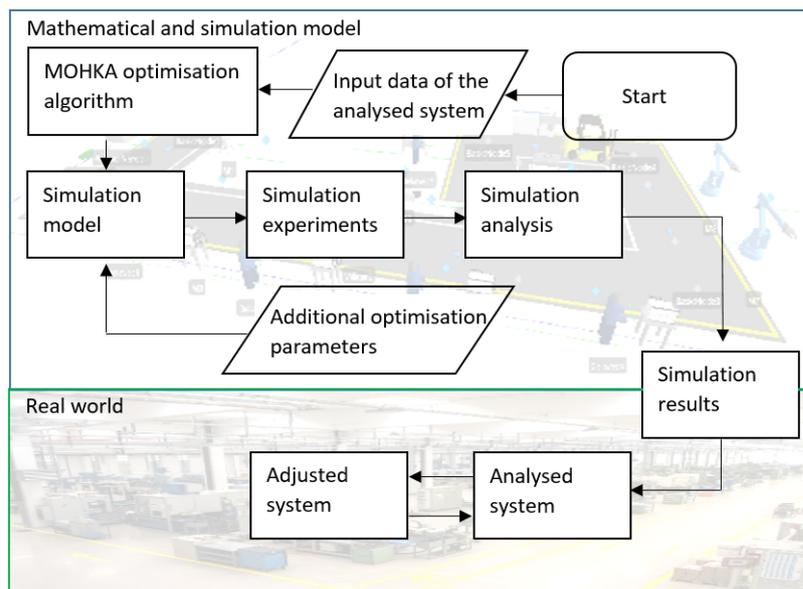


Fig. 3 Block diagram of interactive simulation model

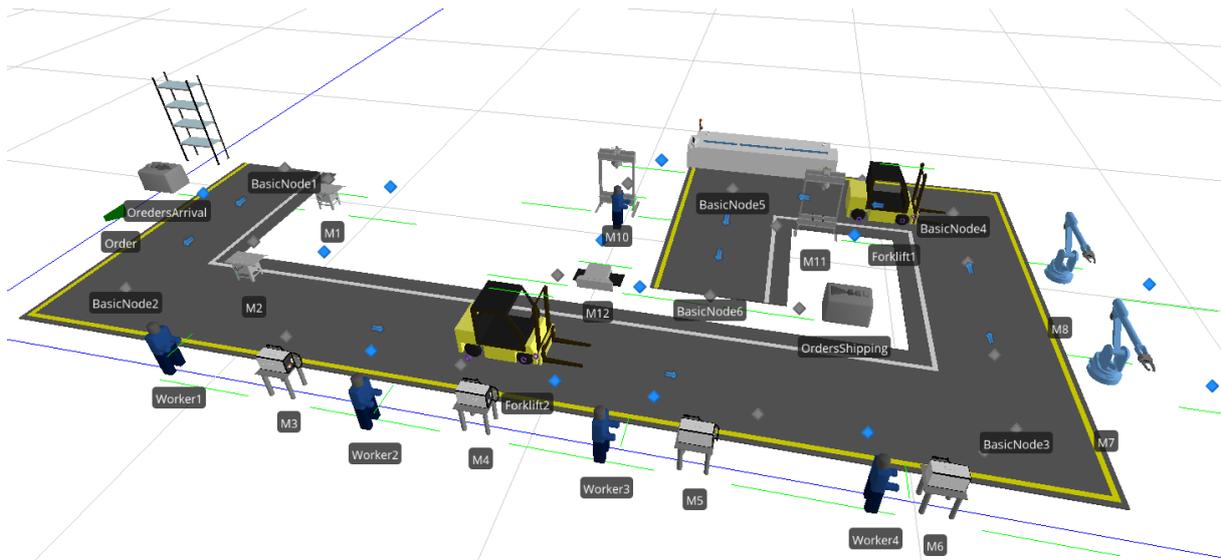


Fig. 4 Simulation model in Simio

Table 6 Optimisation results of MOHKA algorithm transferred to simulation model

Order	Operation	Machine	Start time	Finish time	Machine sequence
J_1	$O_{1,1}$	M_1	0	1	1
	$O_{1,2}$	M_1	1	5	2
	$O_{1,3}$	M_{10}	5	8	2
J_2	$O_{2,1}$	M_{10}	0	4	1
	$O_{2,2}$	M_7	4	5	1
	$O_{2,3}$	M_6	5	8	2
J_3	$O_{3,1}$	M_8	0	3	1
	$O_{3,2}$	M_5	3	5	1
	$O_{3,3}$	M_8	5	7	2
J_4	$O_{4,1}$	M_9	0	1	1
	$O_{4,2}$	M_6	1	5	1
	$O_{4,3}$	M_4	5	6	1
J_5	$O_{5,1}$	M_3	0	4	1
	$O_{5,2}$	M_3	4	7	2
	$O_{5,3}$	M_4	7	8	2

The results of the optimisation algorithm shown in Table 6 are transferred automatically to the Simio software environment via the communication interface MATLAB, Excel, Simio [37]. The transmission of optimisation results allows further evaluation of the results using a simulation model of a real world production system. The MOHKA algorithm allocated orders' operations optimally to individual machines, and transferred these data to the Simio simulation environment. When performing simulation experiments in a simulation environment, a limitation occurs in the decision logic of the simulation environment. Simio's simulation environment has an integrated decision logic for order sequencing, which, in a lot of cases, makes it impossible to follow the order sequence given as the solution of the EC optimisation algorithms. In order to eliminate the integrated decision logic of the simulation environment, we introduced a new functional dependency that defines the sequence of execution of individual orders on the assigned machine.

In Table 6, the row Machine sequence works by assigning the MOHKA optimisation numerical results of the order sequence and the matching numerical results in Table 6 to the row Machine sequence that defines the sequence of operations according to the MOHKA optimisation solution. Example: According to the MOHKA solution (Fig. 5, top chart), the $O_{1,1}$ operation is first performed on the M_1 machine, followed by the operation $O_{1,2}$. The first operation $O_{1,1}$ to be performed on machine M_1 is assigned to this operation consecutive number one, which determines the sequence of execution on the machine. A sequence number two is assigned to operation $O_{1,2}$,

by the above method. The presented approach defines values for the entire sequence of the Kacem 5×10 dataset. The solutions of the approach are shown in Table 6. The smaller the assigned value of the variable Machine sequence is, the sooner it will be executed on the assigned machine. The presented method enables robust, smooth data transfer between the software environments of the optimisation algorithm and the simulation model. With the presented approach, we bypassed the decision logic of the simulation environment, and we could use our own optimisation algorithm to determine the optimum orders' sequence. Fig. 5 shows the Gantt charts' solutions of the optimisation algorithm (top chart) and simulation model (lower chart). The Gantt charts are the same, which proved the high capability of the presented method to integrate the EC method decision logic into the existing simulation environment.

The presented approach emphasises the advantage of a modular adaptive design that allows the simulation model to be adapted to a wide range of multi-objective optimisation problems. The simulation model can be adjusted, upgraded or replaced as needed by any part of the two-part structure. Replacing or upgrading individual parts of the presented approach makes the presented method sustainable.

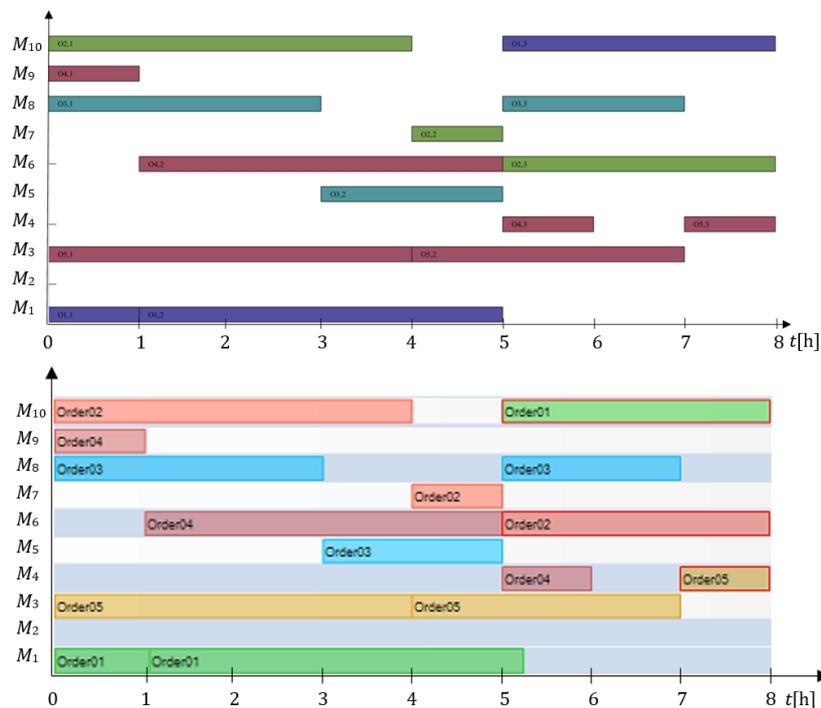


Fig. 5 Gantt chart of the optimisation algorithm (top chart) and simulation model results (lower chart)

4. Conclusion

The presented research work links our own developed EC method of the MOHKA algorithm with the interactive simulation model. The transfer of optimisation results to a simulation environment can, in many cases, represent limitations in the use of its own decision logic. The presented research results have answered the main research question related to the limitation regarding interactivity of mathematical and simulation modelling. The research problem of a multi-objective FJSSP optimisation problem scheduling was identified initially. Identifying an NP-hard optimisation problem requires the use of advanced EC methods. The proposed MOHKA algorithm shows an example of solving the test Kacem [16] and Brandimarte [17] benchmarks. Fifteen datasets were divided into three difficulty groups. According to the proposal of researchers [25], the division of optimisation problems into three levels of difficulty enables a detailed evaluation of the optimisation results. Optimisation results are evaluated numerically and graphically, and a comparative analysis was performed between MOHKA, MOPSO and BBMOPSO. The advantages and limitations of the individual optimisation results were defined [14], which show

the characteristics of the individual algorithm with respect to the corresponding mathematical structure. Based on the satisfactory optimisation results, the presented MOHKA method is suitable for optimisation results' interactivity between the mathematical and simulation models. The interactive method shown gives the advantage of transferring optimisation results via a simulation model to a real environment. The importance of transferring the MOHKA algorithm optimisation results to a real world environment demonstrates the high degree of the proposed method's applicability in complex manufacturing systems supported by the Industry 4.0 concept. The graphical results demonstrate the reliability and robustness of the proposed approach, which will be expanded further by modelling and devaluing the flexibility parameter and its dependence on the production cost-time profile. An adequate time justified profile is key in ensuring sustainable production systems.

The proposed interactive method represents an advanced, flexible and effective link between mathematical and simulation modelling. The open architectural model allows the extension and application of the method to various optimisation problems for both service and production systems. Further research in the field of Mathematical and Simulation Modelling of flexibility parameters in high-mix low-volume production systems will present the optimum production systems' scheduling importance.

Acknowledgement

The authors gratefully acknowledge the support of the Slovenian Research Agency (ARRS), Research Core Funding No. P2-0190.

References

- [1] Schwab, K. (2017). *The fourth industrial revolution*, World Economic Forum, New York, USA.
- [2] Li, Y., Yao, X., Zhou, J. (2016). Multi-objective optimization of cloud manufacturing service composition with cloud-entropy enhanced genetic algorithm, *Strojniški Vestnik – Journal of Mechanical Engineering*, Vol. 62, No. 10, 577-590, doi: [10.5545/sv-jme.2016.3545](https://doi.org/10.5545/sv-jme.2016.3545).
- [3] Chaudhry, I.A., Usman, M. (2017). Integrated process planning and scheduling using genetic algorithms, *Tehnički Vjesnik – Technical Gazette*, Vol. 24, No. 5, 1401-1409, doi: [10.17559/TV-20151121212910](https://doi.org/10.17559/TV-20151121212910).
- [4] Huang, J., Süer, G.A. (2015). A dispatching rule-based genetic algorithm for multi-objective job shop scheduling using fuzzy satisfaction levels, *Computers & Industrial Engineering*, Vol. 86, 29-42, doi: [10.1016/j.cie.2014.12.001](https://doi.org/10.1016/j.cie.2014.12.001).
- [5] Mitchell, M. (1998). *An introduction to genetic algorithms*, MIT press, Cambridge, England.
- [6] Hao, X., Gen, M., Lin, L., Suer, G.A. (2017). Effective multiobjective EDA for bi-criteria stochastic job-shop scheduling problem, *Journal of Intelligent Manufacturing*, Vol. 28, No. 3, 833-845, doi: [10.1007/s10845-014-1026-0](https://doi.org/10.1007/s10845-014-1026-0).
- [7] Meolic, R., Brezočnik, Z. (2018). Flexible job shop scheduling using zero-suppressed binary decision diagrams, *Advances in Production Engineering & Management*, Vol. 13, No. 4, 373-388, doi: [10.14743/apem2018.4.297](https://doi.org/10.14743/apem2018.4.297).
- [8] Wang, H., Wang, W., Sun, H., Cui, Z., Rahnamayan, S., Zeng, S. (2017). A new cuckoo search algorithm with hybrid strategies for flow shop scheduling problems, *Soft Computing*, Vol. 21, No. 15, 4297-4307, doi: [10.1007/s00500-016-2062-9](https://doi.org/10.1007/s00500-016-2062-9).
- [9] Vujica Herzog, N., Buchmeister, B., Beharic, A., Gajsek, B. (2018). Visual and optometric issues with smart glasses in Industry 4.0 working environment, *Advances in Production Engineering & Management*, Vol. 13, No. 4, 417-428, doi: [10.14743/apem2018.4.300](https://doi.org/10.14743/apem2018.4.300).
- [10] Zhang, H., Liu, S., Moraca, S., Ojstersek, R. (2017). An effective use of hybrid metaheuristics algorithm for job shop scheduling problem, *International Journal of Simulation Modelling*, Vol. 16, No. 4, 644-657, doi: [10.2507/IJSIMM16\(4\)7.400](https://doi.org/10.2507/IJSIMM16(4)7.400).
- [11] Fu, H.C., Liu, P. (2019). A multi-objective optimization model based on non-dominated sorting genetic algorithm, *International Journal of Simulation Modelling*, Vol. 18, No. 3, 510-520, doi: [10.2507/IJSIMM18\(3\)CO12](https://doi.org/10.2507/IJSIMM18(3)CO12).
- [12] Gotlih, J., Brezocnik, M., Balic, J., Karner, T., Razborsek, B., Gotlih, K. (2017). Determination of accuracy contour and optimization of workpiece positioning for robot milling, *Advances in Production Engineering & Management*, Vol. 12, No. 3, 233-244, doi: [10.14743/apem2017.3.254](https://doi.org/10.14743/apem2017.3.254).
- [13] Pantić, M., Đorđević, A., Erić, M., Mitrović, S., Babić, M., Džunić, D., Stefanović, M. (2018). Application of artificial neural network in biotribological research of dental glass ceramic, *Tribology in Industry*, Vol. 40, No. 4, 692-701, doi: [10.24874/ti.2018.40.04.15](https://doi.org/10.24874/ti.2018.40.04.15).
- [14] Ojstersek, R., Zhang, H., Liu, S., Buchmeister, B. (2018). Improved heuristic kalman algorithm for solving multi-objective flexible job shop scheduling problem, *Procedia Manufacturing*, Vol. 17, 895-902, doi: [10.1016/j.promfg.2018.10.142](https://doi.org/10.1016/j.promfg.2018.10.142).

- [15] Rupnik, B., Nardin, R., Kramberger, T. (2019). Discrete event simulation of hospital sterilization logistics, *Tehnički Vjesnik – Technical Gazette*, Vol. 26, No. 5, 1486-1491, doi: [10.17559/TV-20180614102011](https://doi.org/10.17559/TV-20180614102011).
- [16] Kacem, I., Hammadi, S., Borne, P. (2002). Pareto-optimality approach for flexible job-shop scheduling problems: Hybridization of evolutionary algorithms and fuzzy logic, *Mathematics and Computers in Simulation*, Vol. 60, No. 3-5, 245-276, doi: [10.1016/S0378-4754\(02\)00019-8](https://doi.org/10.1016/S0378-4754(02)00019-8).
- [17] Mastrolilli, M., Gambardella, L.M. (2000). Effective neighbourhood functions for the flexible job shop problem, *Journal of Scheduling*, Vol. 3, No. 1, 3-20, doi: [10.1002/\(SICI\)1099-1425\(200001/02\)3:1<3::AID-IOS32>3.0.CO;2-Y](https://doi.org/10.1002/(SICI)1099-1425(200001/02)3:1<3::AID-IOS32>3.0.CO;2-Y).
- [18] Mostaghim, S., Teich, J. (2003). Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO), In: *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, Indianapolis, USA, 26-33, doi: [10.1109/SIS.2003.1202243](https://doi.org/10.1109/SIS.2003.1202243).
- [19] Zhang, Y., Gong, D.-W., Ding, Z. (2012). A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch, *Information Sciences*, Vol. 192, 213-227, doi: [10.1016/j.ins.2011.06.004](https://doi.org/10.1016/j.ins.2011.06.004).
- [20] Cajzek, R., Klanšek, U. (2016). Mixed-integer nonlinear programming based optimal time scheduling of construction projects under nonconvex costs, *Tehnički Vjesnik – Technical Gazette*, Vol. 23, No. 1, 9-18, doi: [10.17559/TV-20140108112928](https://doi.org/10.17559/TV-20140108112928).
- [21] Becker, C., Scholl, A. (2009). Balancing assembly lines with variable parallel workplaces: Problem definition and effective solution procedure, *European Journal of Operational Research*, Vol. 199, No. 2, 359-374, doi: [10.1016/j.ejor.2008.11.051](https://doi.org/10.1016/j.ejor.2008.11.051).
- [22] Graham, R.L., Lawler, E.L., Lenstra, J.K., Kan Rinnooy, A.H.G. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey, *Annals of Discrete Mathematics*, Vol. 5, 287-326, doi: [10.1016/S0167-5060\(08\)70356-X](https://doi.org/10.1016/S0167-5060(08)70356-X).
- [23] Lin, L., Gen, M. (2018). Hybrid evolutionary optimisation with learning for production scheduling: State-of-the-art survey on algorithms and applications, *International Journal of Production Research*, Vol. 56, No. 1-2, 193-223, doi: [10.1080/00207543.2018.1437288](https://doi.org/10.1080/00207543.2018.1437288).
- [24] Miettinen, K.M. (2012). *Nonlinear multiobjective optimization*, Springer, New York, USA.
- [25] Deb, K., Agrawal, S., Pratap, A., Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II, In: Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J.J., Schwefel, H.-P. (eds), *Parallel Problem Solving from Nature PPSN VI, Lecture Notes in Computer Science*, Vol. 1917, Springer, Berlin, Germany, 849-858, doi: [10.1007/3-540-45356-3_83](https://doi.org/10.1007/3-540-45356-3_83).
- [26] Deb, K., Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints, *IEEE Transactions on Evolutionary Computation*, Vol. 18, No. 4, 577-601, doi: [10.1109/TEVC.2013.2281535](https://doi.org/10.1109/TEVC.2013.2281535).
- [27] Hwang, C.-L., Masud, A.S.M. (1979). *Multiple objective decision making – Methods and applications: A state-of-the-art survey*, Springer-Verlag, Berlin, Heidelberg, New York, doi: [10.1007/978-3-642-45511-7](https://doi.org/10.1007/978-3-642-45511-7).
- [28] Hassanzadeh, H.R., Rouhani, M. (2010). A multi-objective gravitational search algorithm, In: *2nd International Conference on Computational Intelligence*, Liverpool, United Kingdom, 7-12, doi: [10.1109/CICSyN.2010.32](https://doi.org/10.1109/CICSyN.2010.32).
- [29] Ojstersek, R., Zhang, H., Palcic, I., Buchmeister, B. (2017). Use of heuristic Kalman algorithm for JSSP, In: Andraš. A. (ed.), *XVII International scientific conference on industrial systems*, Novi Sad, Faculty of Technical Sciences, Department for Industrial Engineering and Management, Novi Sad, Serbia, 72-77.
- [30] Demir, Y., Kürşat İşleyen, S. (2013). Evaluation of mathematical models for flexible job-shop scheduling problems, *Applied Mathematical Modelling*, Vol. 37, No. 3, 977-988, doi: [10.1016/j.apm.2012.03.020](https://doi.org/10.1016/j.apm.2012.03.020).
- [31] Chiang, T.-C., Lin, H.-J. (2013). A simple and effective evolutionary algorithm for multiobjective flexible job shop scheduling, *International Journal of Production Economics*, Vol. 141, No. 1, 87-98, doi: [10.1016/j.ijpe.2012.03.034](https://doi.org/10.1016/j.ijpe.2012.03.034).
- [32] Prester, J., Buchmeister, B., Palčić, I. (2018). Effects of advanced manufacturing technologies on manufacturing company performance, *Strojniški Vestnik – Journal of Mechanical Engineering*, Vol. 64, No. 12, 763-771, doi: [10.5545/sv-jme.2018.5476](https://doi.org/10.5545/sv-jme.2018.5476).
- [33] Lu, Y. (2017). Industry 4.0: A survey on technologies, applications and open research issues, *Journal of Industrial Information Integration*, Vol. 6, 1-10, doi: [10.1016/j.jiii.2017.04.005](https://doi.org/10.1016/j.jiii.2017.04.005).
- [34] Ojstersek, R., Buchmeister, B. (2017). Use of simulation software environments for the purpose of production optimization, In: *Proceedings of the 28th DAAAM International Symposium on Intelligent Manufacturing and Automation*, Zadar, Croatia, 750-758, doi: [10.2507/28th.daaam.proceedings.106](https://doi.org/10.2507/28th.daaam.proceedings.106).
- [35] Joines, J.A., Roberts, S.D. (2015). *Simulation modeling with SIMIO: A workbook*, 4th Edition, CreateSpace Independent Publishing Platform, Sewickley, USA.
- [36] Yang, W., Takakuwa, S. (2017). Simulation-based dynamic shop floor scheduling for a flexible manufacturing system in the industry 4.0 environment, In: *Proceedings of the 2017 Winter Simulation Conference*, Las Vegas, USA, 3908-3916, doi: [10.1109/WSC.2017.8248101](https://doi.org/10.1109/WSC.2017.8248101).
- [37] Dehghanimohammadabadi, M., Keyser, T.K. (2017). Intelligent simulation: Integration of SIMIO and MATLAB to deploy decision support systems to simulation environment, *Simulation Modelling Practice and Theory*, Vol. 71, 45-60, doi: [10.1016/j.simpat.2016.08.007](https://doi.org/10.1016/j.simpat.2016.08.007).