Advanced Logistic Systems – Theory and Practice, Vol. 16, No. 1 (2022), pp. 31-40.
https://doi.org/10.32971/als.2022.003

OPTIMISATION OF FLOW SHOP SCHEDULING PROBLEM: SIMULATION SYSTEM VS. EVOLUTIVE SOLVER

IRENEUSZ KACZMAR[1]–TAMÁS BÁNYAI[2]

Abstract: In the Industry 4.0 era the optimisation of production processes become more and more important. A wide range of tools are used to improve the performance of production processes and related operations of the supply chain including purchasing, distribution and inverse processes. These tools have a great impact on the results of design and operation of production processes both technology and logistics point of view. Within the frame of this article, the authors are demonstrating the optimisation potentials of simulation tools and heuristic solvers (optimisers) through the problem of the well-knows permutation flow shop scheduling. They are describing the models and methods used in the case of FlexSim simulation systems and the heuristic solver option of Excel Solver. As the numerical results show, both the simulation model and the heuristic optimisation approach lead to a quasi-optimal solution.

Keywords: *simulation, FlexSim, solver, optimisation, permutation flow-shop*

1. INTRODUCTION

Scheduling of tasks belongs to a large class of problems, considered in various configurations, for which many optimization criteria can be adopted, such as time, cost, shortest path, capacity utilisation and others. The topics discussed are still relevant and described in the literature [1, 2]. A lot of cited works are focusing only on the problem of scheduling of production processes focusing on technology, but in general, this problem may concern most logistic processes. The purpose of this work is to compare the solution of the problem of scheduling tasks in the simulation model in FlexSim and an evolutive solver tool. The optimization criterion is the minimization of the total execution time. The rapid development of computer technology and artificial intelligence has enabled the widespread use of general-purpose optimizers. As a consequence of the computer simulations, the estimated execution time was obtained, with different configurations of task scheduling for the process. In a broader context, the presented solution can be used in many fields of optimization, including:

- study of the execution time of any process;
- optimization of customer service time;
- examination of the operator's work efficiency;
- optimization of the order processing with different times of performance of individual operations, e.g. order picking for many customers, including transport with a complex number of sub-tasks, and others.

Based on this fact, the paper is organized as follows. Section 2 presents a short knowledge overview regarding permutation flow-shop scheduling. Section 3 presents the materials and methods, while section 4 focuses on the results and conclusions.

[1] Associate professor, Institute of Technical Sciences, The East European State Higher School in Przemysl
i.kaczmar@pwsw.eu
[2] Associate professor, Institute of Logistics, University of Miskolc
alttamas@uni-miskolc.hu

2. THEORETICAL BACKGROUND

In production and logistics systems, the optimal schedule is determined primarily by two basic elements:

- the order of processing details / tasks and
- technological route.

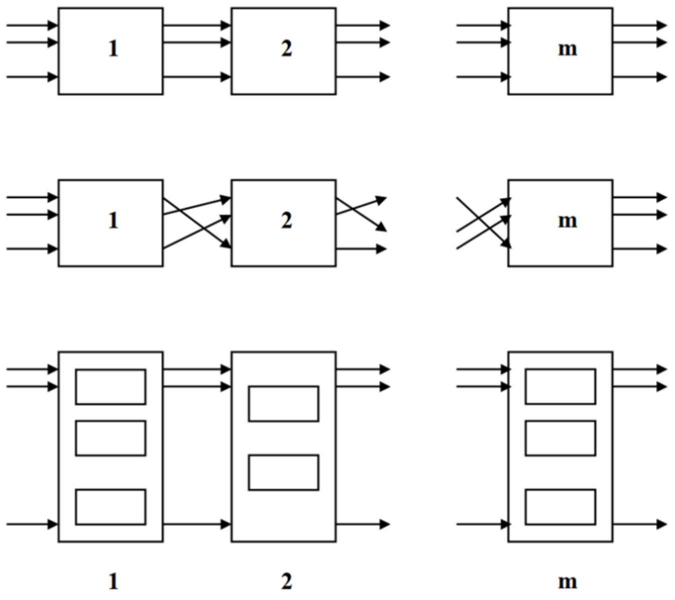

Figure 1. Structures of flow-shop production systems: flow (F), permutation (PF), non-permutation with parallel machines at the station [3]

The basic flow systems of objects or tasks in the production processes are shown in Figure 1 and in Figure 2 [3]. Identification of flow systems:

- **F:** flow shop (Figure 1), in which all tasks have the same technological route and require service at all stations, and each station requires the determination of an appropriate sequence of task entry,
- **PF:** permutation flow-shop (Figure 1), which has the same assumptions as **F** with an additional requirement, that the order of operation of tasks on all machines is the same (consistent with the order of entering tasks into the system),
- **J:** job-shop (Figure 2) in which different tasks may have different (in terms of the number and order of visiting positions) technological routes,
- **G:** general shop, where each task is a single operation, and technological dependencies are data described any graph,
- **I:** parallel shop, in which each task is a single operation and all operations are performed on exactly one of several parallel machines,
- **O**: open flow shop, in which all operations for task have to be performed, but the technological sequence of operations in the task is not specified.

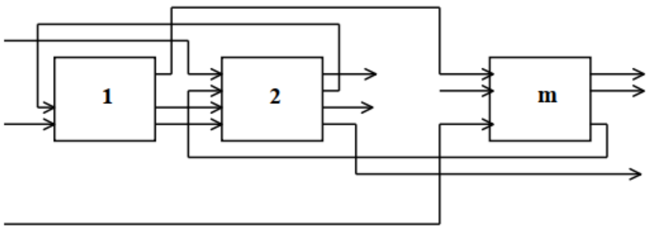

Figure 2. Job-shop manufacturing system example [3]

If the process route which is determined by the production technology cannot be changed, then the only possible form of optimization is the ordering of tasks. Permutation is the setting of the elements of a given set in a certain order. The number of permutations determines how many ways we can queue the elements of the set. Thus the number of all permutations of an n-element set can be written as follows with (1):

$$P_n = n! \tag{1}$$

If the sequence of tasks on individual machines is the same, then this case is called a permutation flow problem. The decision variable is the order in which details enter the system, according to the difficulty of the written formula (1). For such a flow system with a fixed technological route, the difficulty of scheduling tasks will bring for 5 types of details 5! = 120 setting options, for 6 types of details 6! = 720 settings options, for 7 types of details 7! = 5040 settings, and 8 types of details we already have over 40 thousand possibilities to set the order of parts to be processed.

The solution of this problem is to determine the order of tasks on the first machine, because the order of processing on subsequent machines is predetermined by the technological route. According to [4], we are looking for a permutation (π) which that will minimize the total flow time of all tasks (2). The maximum time for completion of all tasks (C_{max}) is referred to in the literature as makespan (4). The discussed problem belongs to NP-hard problems and approximate algorithms are used to solve it.

$$\Phi(\pi) = \max_{1 \le i \le n}(f_i(\pi)) \tag{2}$$

where $f_i(\pi)$ is the time to complete the i-th task for the ranking π.

General described in the literature [5] the flow shop problem with blocking has some assumptions. In the system there are *M* – machines on which to perform *J* – jobs consisting of one or more operations – O_j [6]:

- collection of machines: $M = \{M_1, M_2, M_3, ..., M_k, ... M_m\}$,
- a set of production jobs: $J = \{J_1, J_2, J_3, ..., J_j, ..., J_n\}$,
- every task $J_j \in J$ consists of *m* operations from the set: $O_j=\{O_{j1}, O_{j2}, O_{j3}... O_{jk}... O_{jm}\}$, operation $O_{jk} \in O_j$ corresponds to the processing of the task *j* on the machine *k* during uninterrupted time $p_{jk} > 0$ (collection of durations p_j operations from the set $O_j : p_j = \{p_{j1}, p_{j2}, p_{j3},..., p_{jk},..., p_{jm}\}$),
- each machine can perform only one operation at a time,
- there are no buffers in the system for storing items (tasks), that can move from the current machine to the next only when it is free, so the task *j* after the processing of the operation is completed O_{jk}, cannot leave the machine *k*, as long as the

machine *k+1* will not be free, if the machine k+1 is busy the system is locked, where: $k = 1, 2, \ldots, m - 1, j \in J$.

A variable decision in this problem is the order in which tasks are performed on machines. It is described by a set of *m*-permutations: $\pi = (\pi_1, \pi_2, \ldots, \pi_i, \ldots, \pi_m)$. The number of sets of all possible permutations for this problem is $(n!)^m$. Let Π denote the set of all such permutations, then such a permutation should be found. $\pi^* \in \Pi$, that:

$$\mathrm{C}_{max}(\pi^*) = \min_{\pi \in \Pi}\left(\mathrm{C}_{max}(\pi)\right) \tag{3}$$

where: $\mathrm{C}_{\max}(\pi)$ is the time needed to perform all jobs on machines in the order given by the π. $\mathrm{C}_{\max}$ is the deadline for completing all tasks:

$$\mathrm{C}_{max} = \max_{1 \le i \le n}\left(\mathrm{C}_j\right) \tag{4}$$

where *n* is the number of tasks, C_j is the deadline for completing task *j*.

Suboptimal methods for task scheduling known in the literature include: S. M. Johnson's algorithm, combinatorial methods [7,8], simulation methods, graphoanalytical method, Palmer's algorithm [9]. The review of algorithms supporting the planning and scheduling of tasks was made by [10-12]. Application of the Palmer's algorithm has been described among others in works [8, 13,14]. The colony algorithm is inspired by the behaviour of real ants that can be used to solve combinatorial optimization problems. Ant behaviour is used to construct graph solutions for flow problems which are then corrected by local search algorithms [15]. The performance improvement of production processes using simulation is discussed in [16].

3. MATERIALS AND METHODS

A computational example using the general purpose metaheuristic optimizer OptQuest has been implemented in the simulation model. The main engine of this optimizer is based on the distributed search methodology *(scatter search)* combined with a mechanism *(tabu search)*. OptQuest Optimizer was developed by F. Glovera, J.P. Kelly'ego i M. Laguna in University of Colorado. The first version was developed to optimize the simulation of discrete events modelled with Micro Saint 2.0 [17]. Currently this software is an optimization engine added to simulation packages such as: Arena, Promodel, Simul8, Simio and others. Scatter search is the main and the first optimization technology implemented in OptQuest. Over the years, however, many technologies have emerged for both prediction and optimization have been added to the optimizer engine. There are several options and settings to choose from to change the behaviour of the goal solving process. In subsequent versions of the program, new predictive functions were implemented for the purpose of the optimization problem being solved. Predictive models in the engine have the dual purpose of helping to establish search directions as well as to estimate the proper value of the objective function. A more detailed description of the optimizer's operation can be found in the literature [18].

Our task is to prepare a simulation model to calculate the optimal schedule for machining nine details: $D=\{D_1 \ldots D_9\}$ on eight machines $M = \{M_1, M_2, M_3, M_4, M_5, M_6, M_7, M_8\}$, with a fixed technological route. The number of possible permutations in this case is 9! For collection of machines the technological route of the items was fixed (Table I) in the next steps from S_1 to S_9, an additional step S_{10} with a time of zero is dedicated to the output of the flow element from the system and is not relevant to the calculations performed. Table I shows that each detail is processed in nine steps of the production process. Every detail passes through all machines from M_1 to M_9. The technological route consists of 9 steps for 9 details. A process time matrix was also created t_{ij} (Table II).

Table I.

Technological route for 9 of details

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
D_1	1	2	3	4	5	6	7	8	9	exit from the system
D_2	1	2	3	4	5	6	7	8	9	exit from the system
D_3	1	2	3	4	5	6	7	8	9	exit from the system
D_4	1	2	3	4	5	6	7	8	9	exit from the system
D_5	1	2	3	4	5	6	7	8	9	exit from the system
D_6	1	2	3	4	5	6	7	8	9	exit from the system
D_7	1	2	3	4	5	6	7	8	9	exit from the system
D_8	1	2	3	4	5	6	7	8	9	exit from the system
D_9	1	2	3	4	5	6	7	8	9	exit from the system

Table II.

Process time for each details

	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9
M_1	2	3	6	5	4	1	2	6	7
M_2	4	5	4	4	5	3	3	5	3
M_3	1	6	3	6	3	4	2	4	4
M_4	6	2	1	3	6	4	3	3	5
M_5	5	1	5	1	2	6	4	2	7
M_6	2	4	5	2	3	4	4	5	2
M_7	7	4	2	2	3	5	5	8	3
M_8	6	5	1	3	2	5	5	7	4

Figure 3 shows a block diagram of a computer model for testing system performance in the FlexSim environment. A diagram of the logical connections of the model was developed and the input parameters were introduced, i.e. machining times and technological route. The total time for completing tasks in the permutation schedule will be calculated. In the analyzed case, the order of entry into the system will be changed to minimize the objective function. The results obtained from the simulation experiments will be discussed in the following sections.

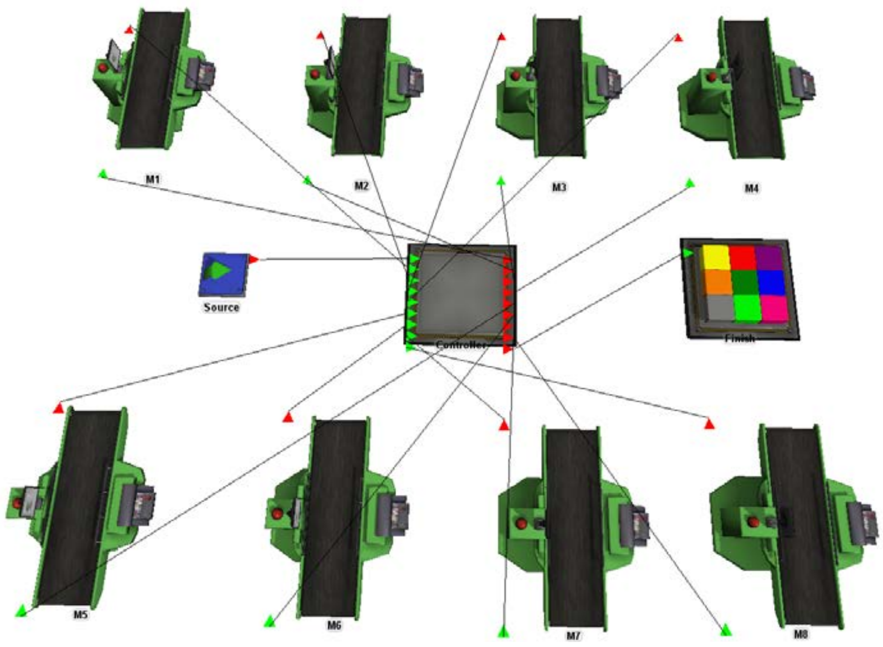

Figure 3. An example of a permutation flow system in a simulation model

The next optimisation approach focuses on the solution of flow-shop scheduling problems with the Excel Solver's evolutive optimiser to validate the results of simulation-based scheduling. In the case of this approach the objective function of the solver model is the minimisation of the timespan required to perform all operations. The decision variable is the optimal sequence of operations, which can be represented by a permutation of operations. The following parameters were used while solving the above described problem [19]:

- Convergence defines the maximum percentage difference in objective values for the top 99% of the population that Solver should allow in order to stop with the message "Solver converged to the current solution." We have chosen for this parameter *c=0.0001*.
- Mutation Rate defines the relative frequency with which some member of the population will be mutated. However, a higher mutation rate can increase the diversity of the population, but we have defined a mutation rate *mr=0.075*.
- Population Size defines the number of potential solutions, which are taken into consideration and managed in the same time. In our solver approach we have defined 100 individuals representing 100 potential solutions as the size of the population: *ps=100*.
- Random seed defines the initial value of the random generator. We have left this value blank, so the random generator uses different random seeds for every optimisation.
- Maximum time without improvement parameter defines the time the user want to run the solver without finding a better solution. We have defined for this parameter *mt=30 s*.

4. Results and discussion

The method of discrete simulation will be used to compare the three variants of scheduling tasks. In a computer environment, the developed model (Fig. 3) was launched for three permutation variants of the technological route of the workpieces *A*, *B*, *C*. Variant V_A - items to be processed enter in the order from the first to the last, variant V_B - items enter from the last to the first, V_C variant - items enter the system in the optimal order:

- $V_A = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$;
- $V_B = \{9, 8, 7, 6, 5, 4, 3, 2, 1\}$;
- $V_C = \{6, 1, 4, 5, 2, 3, 8, 7, 9\}$; *optimal sequence based on simulation (72 time units)*

Based on the conducted simulation experiments Gantt's charts has been drawn, which show the occupation and idleness of machines for individual variants and the total execution time (Figures 4-6). In the Gantt chart, black means that the machine is idle, while white means that the machine is busy.

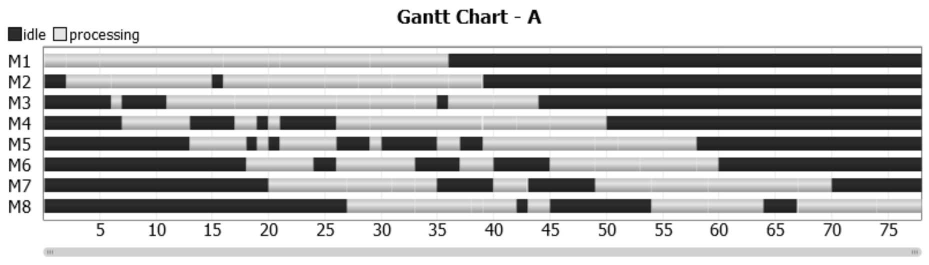

Figure 4. Gantt charts show the use of machines in various variant A

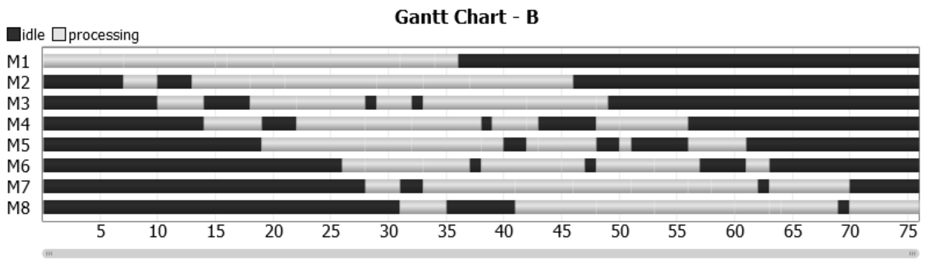

Figure 5. Gantt charts show the use of machines in various variant B

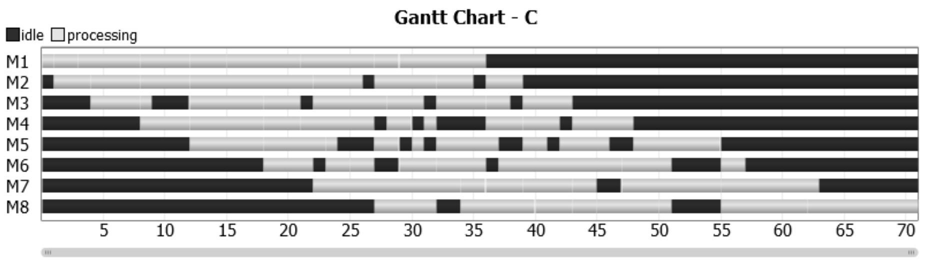

Figure 6. Gantt charts show the use of machines in various variant C

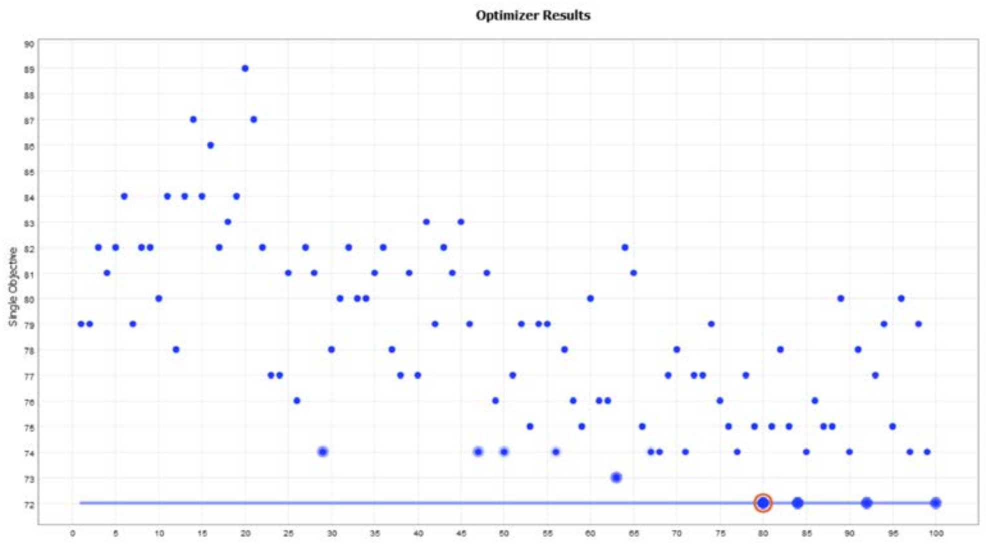

Figure 7. Results obtained from the optimizer

As demonstrated by the simulations carried out, variant C turned out to be optimal with a total execution time of 72 time units. For variant B - 80 units were obtained, for variant A - 78 time units. From the possible (9!) set of permutations, the optimizer chose the best 100 solutions, shown in Figure 7. As a result of the conducted experiments, reliable results ranging from 89 to 72 time units were obtained. The real working time of the optimizer - 9 seconds, the number of permissible permutations – 100. Hardware Intel(R) Core(TM) i9-9900KF CPU 3.60 GHz with 16GB RAM, RTX 2070 Super.

In conclusion, it can be assumed that the algorithms available today in the commonly used optimizers (including the tested optimizer OptQuest) are very good at scheduling tasks. And simulation programs, among others FlexSim, Arena, Promodel and others allow you to quickly build and check the performance of even complex models of production systems that can be easily adapted to other fields to optimize tasks.

The second optimization approach was the evolutive solver, which found a better solution. As Figure 8 shows, the optimal sequence required 64 time units, which represents 90% of the solution of the simulation-based approach. Blue and orange represent the production, while gray is for idle time.

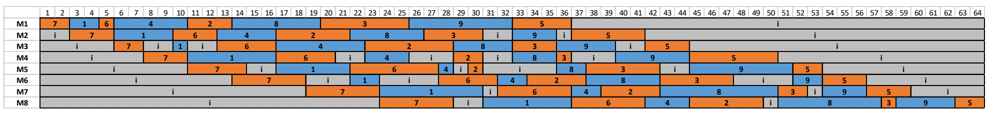

Figure 8. Results obtained from the evolutive solver

5. SUMMARY

Simulation and optimisation represent two important approach to improve the efficiency of production and logistics processes. These tools are important both for in-plant problems and for supply chain solutions [20]. Within the frame of this article the authors demonstrated the potentials of simulation and heuristic-based optimizers in the case of permutation flow-shop problems. As the computational results and simulation studies show, both the simulation and the heuristic optimizer resulted a quasi-optimal solution. The future research direction is to make more sophisticated comparisons including more potential solution methodologies and tools [21, 22] and include more simulation based approaches [23].

REFERENCES

[1] Walker, R. A. & Samit, Ch. (1995). Introduction to the scheduling problem. *IEEE Design & Test of Computers*, **12**(2), 60-69. https://doi.org/10.1109/54.386007

[2] Koulamas, Ch. & Kyparisis, G. (2022). Flow shop scheduling with two distinct job due dates. *Computers & Industrial Engineering*, **163**, https://doi.org/10.1016/j.cie.2021.107835

[3] Smutnicki, Cz. (2012). *Algorytmy szeregowania zadań*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław.

[4] Chmiel, W., Kadłuczka, P. & Packanik, G. (2011). *Zastosowanie algorytmów rojowych w rozwią-zywaniu zagadnień permutacyjnych*. Automatyka, AGH im. Stanisława Staszica w Krakowie 15.

[5] Grabowski, J. & Pempera, J. (2007). The permutation flow shop problem with blocking. A tabu search approach. *Omega*, **35**(3), 302-311, https://doi.org/10.1016/j.omega.2005.07.004

[6] Pempera, J. (2002). *Algorytmy szeregowania zadan w pewnym dyskretnym procesie produkcyjnym*. Rozprawa doktorska (28.02.2001) Politechnika Wrocławska, Instytut Cybernetyki Technicznej

[7] Błażewicz, J. (1987). Selected topics in scheduling theory. *North-Holland Mathematics Studies*, **132**, 1-59, https://doi.org/10.1016/S0304-0208(08)73231-6

[8] Pawłowicz, L. (2005). *Ekonomika przedsiębiorstw*. Zagadnienia wybrane. Gdańsk: ODDK.

[9] Skorupski, J. & Kwasiborska, J. (2014). Metody szeregowania zadań jako narzędzie rozwiązywania problemu sekwencjonowania samolotów. *Prace Naukowe Politechniki Warszawskiej, Transport*, **101**, 55-62.

[10] Hissashi, H. & Nagano, M. (2021). Optimizing distributed no-wait flow shop scheduling problem with setup times and maintenance operations via iterated greedy algorithm. *Journal of manufacturing Systems*, **61**, 592-612, https://doi.org/10.1016/j.jmsy.2021.10.005

[11] Crandall, K. C. (1976). Probabilistic time scheduling. *Journal of the Construction Division*, **102**(3), 415-423, https://doi.org/10.1061/JCCEAZ.0000619

[12] Vinod, V. & Sridharan, R. (2011). Simulation modeling and analysis of due-date assignment methods and scheduling decision rules in a dynamic job shop production system. *International Journal of Production Economics*, **129**(1), 127-146. https://doi.org/10.1016/j.ijpe.2010.08.017

[13] Hong, T. P., Wang, T. T. & Wang, S. L. (2001). A palmer-based continuous fuzzy flexible flow-shop scheduling algorithm. *Soft Computing*, **5**(6), 426-433. https://doi.org/10.1007/s005000100109

[14] Hundal, T. S. & Rajgopal, J. (1998). An extension of Palmer's heuristic for the flow shop scheduling problem. *International Journal of Production Research*, **26**, 1119-1124. https://doi.org/10.1080/00207548808947922

[15] Stützle, Th. (1998). An ant approach to the flow shop problem. *Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing*. Vol. **3**. 1560-1564.

[16] Godó, T., Veres, P. & Tóth, N. (2020). Gyártósori anyagellátás hatékonyságnövelése modelltervezéssel és szimuláció alkalmazásával. *Multidiscipinary Sciences*, **10**(2), 260-271. https://doi.org/10.35925/j.multi.2020.2.31

[17] Glover, F., Kelly, J. P. & Laguna, M. (1996). New Advances and Applications of Combining Simulation and Optimization. *Proceedings of the 1996 Winter Simulation Conference*, J. M.

Charnes, D. J. Morrice, D. T. Brunner, and J. J. Swain (eds.), pp. 144-152. https://doi.org/10.1145/256562.256595

[18] Laguna, M. (2011). *OptQuest*. *Optimization of Complex Systems.* Computer Science. OPTTEK SYSTEMS, INC.

[19] *Frontline Solvers* (2022). Excel Solver - change options for evolutionary solving method. https://www.solver.com/excel-solver-change-options-evolutionary-solving-method. Accessed: 14/05/2022

[20] Nagy, G., Bányai, Á. & Illés, B. (2022). Examining the Efficiency of Supply Chains. Advanced *Logistic Systems: Theory and Practice,* **15**(2), 28-34, https://doi.org/10.32971/als.2021.010

[21] Veres, P., Illés, B. & Landschützer, C. (2018). Supply Chain Optimization in Automotive Industry: A Comparative Analysis of Evolutionary and Swarming Heuristics. *Lecture Notes in Mechanical Engineering*, **2018**. Springer, Cham. https://doi.org/10.1007/978-3-319-75677-6_57

[22] Kaczmar, I. (2019). *Komputerowe modelowanie i symulacje procesów logistycznych w środowisku FlexSim*, Wydawnictwo Naukowe PWN, Warszawa.

[23] Molnár, Z., Tamás, P., & Illés, B. (2022). Building a Reusable Data Model to Support Systematic Layout Planning with Discrete Event Simulation of Flexible Manufacturing Systems. *Hungarian Journal of Industry and Chemistry*, **49**(2), 97–100, https://doi.org/10.33927/hjic-2021-29