Minimization of the Number of Employees in Manufacturing Cells

Wojciech Bożejko(✉), Jarosław Pempera, and Mieczysław Wodecki

Faculty of Electronics, Wrocław University of Technology, Janiszewskiego 11-17,
50-372 Wrocław, Poland
{wojciech.bozejko,jaroslaw.pempera,mieczyslaw.wodecki}@pwr.edu.pl

Abstract. In the paper we consider the problem of scheduling of technological operations implementation in productions cells in a company which produce steel structures of car seats. We propose an optimization algorithm based on the Branch and Bound method which determines minimal number of team members which operating production cells maintaining the maximum efficiency of the cells. The usefulness of the algorithm for practical purposes has been verified on real data.

Keywords: Optimization · Production cell · No-store constraint
Cyclic scheduling

1 Introduction

Robotization and automation of manufacturing systems enables relatively quick adaptation of production plans to the demand generated by contractors. This is particularly important in the case of production in large volumes, large sizes or with large masses. Then, storage costs can be a significant part of the cost of manufacturing of the final product. In this type of companies, a just-in-time or just-in-sequence strategies are used.

In the enterprise at work, the process of manufacturing seats and other car equipment includes product design, computer and real-life testing related to the safety of their use, and the production of finished products. Extensive experience in both designing and production of products allows the company to produce many products for many contractors.

Despite of the significant and continuous development of robots and automation devices, the most important element of the manufacturing system is man. In advanced production systems, its role is often to precisely place processed products in the holders of automatic welding machine modules, to make welding corrections or to weld in hard to reach places, to carry out product quality control and to level and polish the surface of products at various stages of production.

Nowadays a chronic shortage of highly-qualified workforce is observed in many countries. This situation does not change significantly despite an increase in financial incentives for employees. Therefore, effective management of the staff is very important.

M. Graña et al. (Eds.): SOCO'18-CISIS'18-ICEUTE'18, AISC 771, pp. 241–248, 2019.
https://doi.org/10.1007/978-3-319-94120-2_23

2 Manufacturing Cell

The task of a typical production cell is to weld previously embossed metal parts into a raw intermediate or end product. In later stages, it is varnished and packaged. The cell consists of several working places (here called machines). In each of the machines, specific technological activities are performed. Most often these are welding operations performed in welding modules or in a few cases by hand. Welding operations are preceded by the placement of the product in the holder that the operator performs and supervises. The machines in the cell are placed on the perimeter of the straight-angle. Metal parts produced on the press and other parts are delivered in containers placed in a separate place near the working place.

Cells are designed for the production of one product or several types of similar products. The change of production from one product to another requires in some cases the replacement of some tools. Products in the cell are produced in series. In one series, identical products are produced. There are no buffers (temporary storage places) between the positions where the partially processed product could be stored. It is caused by a limited surface allocated for the cell and additional activities and costs related to the operation of buffer devices. The operator may serve several machines, in particular, machines that are physically close to each other. The maximum productivity of the production cell is directly related to the time of performing the longest technological activity.

Cyclic production is required to increase the efficiency and simplify the operation of the operators. In cyclic production, the operator performs most of the working time in the same order. This rule does not apply for a short period at the beginning and the end of the production of series of products.

We want to set a minimum number of cell operators (employees, or – in the future – robots) ensuring production with its maximum efficiency. Determination of such a number is of special importance at the time of high absenteeism of employees or urgent execution of many orders for contractors.

3 Problem Description

The production cell consists of m production machines from the set $M = \{1, 2, \ldots, m\}$. In the cell, cyclically, n products should be made. Each product is processed in each machine in the order of $1, 2, \ldots, m$. Each product therefore generates a production task consisting of m operations. There are no buffers between stations [11,12] (so called *no-store* constraint). The processing time of the job at the position $k \in M$ is $p_k > 0$ [minutes]. The maximum cell productivity is defined as

$$P = 60/\max_{k \in M} p_k \tag{1}$$

pieces per hour.

Each employee is qualified to work in any machine, performs the tasks in a given machine or supervises them throughout the entire processing time in the position. Only one product can be made at a given time, the product can be processed on only one machine and the employee can work on only one machine. Performing tasks on machines cannot be interrupted.

Each technological operation performed by the operator in the system is represented by the pair (j,i), where $j = 1,2,...,n$ denotes the sequence number of the product, and $i \in M$ machine for the cell. Let us denote by α_k the order of performing technological activities by the operator $k = 1,2,...,o$. The sequence of performing all operations by operators in the production cell is described by $\alpha = (\alpha_1,\ldots,\alpha_o)$, $\alpha_k = (\alpha_k(1),\ldots,\alpha_k(n_k))$, where n_k is the number of operations performed by the operator k. Further, let us denote by $S_{j,i}$ and $C_{j,i}$ the start and completion time, respectively, of the task j on a machine i. The schedule of performing tasks for a given sequence α has to fulfill the following constraints:

$$S_{j,i} \geq 0, \quad j = 1,\ldots,n, \quad i = 1,\ldots,m, \tag{2}$$

$$S_{j,i} \geq C_{j,i-1}, \quad j = 1,\ldots,n, \quad i = 2,\ldots,m, \tag{3}$$

$$C_{j,i} = S_{j,i} + p_{j,i}, \quad j = 1,\ldots,n, \quad i = 1,\ldots,m, \tag{4}$$

$$S_{\alpha_k(s)} \geq C_{\alpha_k(s-1)}, \quad s = 2,\ldots,n_k, \quad k = 1,\ldots,o, \tag{5}$$

$$S_{\alpha_k(s)} \geq S_{A(\alpha_k(s-1))}, \quad s = 2,\ldots,n_k, \quad k = 1,\ldots,o, \tag{6}$$

where $A((j,i)) = (j,i+1)$ denotes the technological successor of the operation (j,i). Inequality (2) anchors the task execution schedule at time 0. Inequality (3) means that the start time of the operation of a given task cannot be earlier than the end of the previous operation of this task. Equality (4) means that the operation cannot be interrupted. It results from the inequality (5) that the start time of the s-th operation performed by the operator k can only start after the previous operation has been completed. The limitation "no-store" models the constraint (6) and means that we can start operations only after the job has been released by the previous task, i.e. when the next operation of this task begins its execution.

The order of operations performed by the α operators is feasible if there is a solution of the inequalities (2–6). Checking the feasibility of α and the schedule of performing operations in the order α can be determined in time $O(nm)$ by constructing a directed graph based on the inequalities (2–6) (similar to work [8]).

The sequence α will be called the cyclic sequence if it is constructed on the basis of a cyclic core. The cyclic core consists of cyclically performed m operations for one or more tasks. The order of executing operations from the cyclic core will be denoted by symbolem $\pi = (\pi_1,\ldots,\pi_o)$, $\pi_k = (\pi_k(1),\ldots,\pi_k(\vartheta_k))$, $m = \sum_{s=1}^{o} \vartheta_s$ where ϑ_k is the number of operations performed by the operator k. Then the order α is a core assembly, i.e.

$$\propto = \pi^0 \pi\pi \ldots \pi\pi^*, \tag{7}$$

where π^0 - stands for the preliminary order, $\pi\pi...\pi$ - repeatedly repeating the cyclic core, π^* - the ending order.

The most commonly considered literature in the literature are fully automated production cells in which the transport of products is carried out by an industrial robot [3,6,7,9].

3.1 Example

The production cell consists of $m = 6$ positions and is operated by $o = 4$ operators. The names of technological activities and machining times [in seconds] are given in Table 1. Operators 1, 2, 3 perform only one operation respectively 3, 4 and 6, while operator 4 operations 1, 2 and 5. Consider a cyclic core $\pi = (\pi_1, \ldots, \pi_4)$, in the following form: $\pi_1 = ((1,3))$, $\pi_2 = ((1,4))$, $\pi_3 = ((1,4))$, $\pi_4 = ((5,1), (1,3), (2,3))$.

The system should perform $n = 7$ products. For such a cyclic core, the order π^0 and π^* take the form $\pi^0 = (\pi_4^0)$, where $\pi_4^0 = ((1,1), (2,1), (1,2), (2,2), (1,3), (2,3))$ and $\pi^* = (\pi_1^*, \pi_2^*, \pi_3^*, \pi_4^*)$, where $\pi_1^* = ((5,3), (6,3), (7,3))$, $\pi_2^* = ((5,4), (6,4), (7,4))$, $\pi_3^* = ((5,6), (6,6), (7,6))$, $\pi_4^* = ((5,5), (6,5), (7,5))$.

Table 1. Technological activities performed in the production cell

Position	Operation name	Execution time [sec.]
1	Clinching	45
2	Welding	26
3	Welding	134
4	Manual welding	104
5	Gurtholma welding	34
6	Assembly	104

Figure 1 illustrates the schedule for performing operations in the form of Gantt chart. In the graph: dotted line, the schedule for performing operations resulting from the order π^0 and π^* is marked, the continuous operation schedule resulting from the first and last occurrence of the cyclic core is marked, the

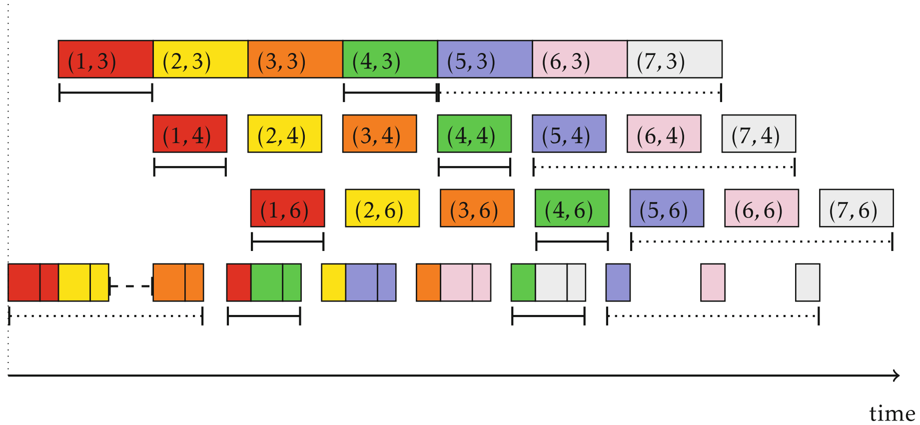

Fig. 1. The schedule of operations performed by operators

Table 2. Schedule of technological operations

Part		Operations					
		1	2	3	4	5	6
1	S_{1i}	0	45	71	205	309	343
	C_{1i}	45	71	205	309	343	447
2	S_{2i}	71	116	205	339	443	477
	C_{2i}	116	142	339	443	477	581
3	S_{3i}	205	250	339	473	577	611
	C_{3i}	250	276	473	577	611	715
4	S_{4i}	343	388	473	607	711	745
	C_{4i}	388	414	607	711	745	849
5	S_{5i}	477	522	607	741	845	879
	C_{5i}	522	548	741	845	879	983
6	S_{6i}	611	656	741	875	979	1013
	C_{6i}	656	682	875	979	1013	1117
7	S_{7i}	745	790	875	1009	1113	1147
	C_{7i}	790	816	1009	1113	1147	1251

dashed line the delay resulting from station blocking due to the lack of interstate buffer. In addition, the start and end times of all operations are summarized in Table 2.

Let us note that the operator 4 supports 3 positions: 1, 2 and 5. Therefore, they should be located nearby, eg stations 1 and 2 should be placed next to each other while 5 in front of positions 1 and 2. The work schedule of this operator is the most complicated, especially at the beginning of work. First, it performs the first two operations of the first task, then the first two operations of the second task, then it must wait for the second position to be released (i.e. until the task 2 in step 1 starts) and perform the first two operations of the third task. Subsequent production activities of the operator 4 are carried out cyclically and consist of three technological operations, successively in stand 5, 1 and 2, and a break related to the wait for the operator 2 to complete the operation.

4 Optimization Algorithm

At the very beginning, note that after a sufficient number of repeats of the cyclic core, the starting moments for the corresponding operations belonging to two successive cores $x-1$ and x are separated by the same period of time T, i.e.

$$S_{x,i} = S_{x-1,i} + T, \quad i = 2, \ldots, m. \tag{8}$$

The period T will be called the *cycle time*. It is easy to see that the production socket works at full productivity (its performance is limited by the time of the longest operation performed) if the cycle time is $\max_{k \in M} p_k$.

The length of the cycle time for the cyclic core π can be determined by generating the order α and determining T in the steady state. Unfortunately, this type of method requires the generation of α based on the large number of core repetitions. Cycle time can be determined much faster using a method similar to that used in [1,2,4,5,10].

In a natural way, the lower limit of the number of operators is FT/T, where $FT = \sum_{s=1}^{m} p_s$ is the time the task flows through the production system, while the upper limit is equal to the number of machines. The minimum number of operators can be determined by searching half of the minimum number of operators for which a schedule with cycle time $T = \max_{k \in M} p_k$ can be constructed.

Due to the small number of operations in the cyclic core, it was decided to construct a complete review algorithm, in which all possible operations assignments to operators and all possible operator execution orders are generated. During the operation of the algorithm, cyclic cores fulfilling the condition

$$\max_{k \in M} \sum_{i=1}^{\vartheta_k} p_{\pi_k(i)} > T, \tag{9}$$

were not generated.

Of course, among the cores that meet this condition may be those for which it is impossible to construct a cyclic schedule with a cycle time T. The algorithm ends when the core is found with the cycle time T or after generating all cyclic cores for a given number of operators.

5 Computational Experiments

Computer research was carried out on 20 products produced in the company. The following data was collected for each product:

1. m - the number of technological operations,
2. $FT = \sum_{s=1}^{m} p_s$- time of task flow through the production system,
3. $T = \max_{k \in M} p_k$ – cycle time.

Next, the minimum number of employees NO was determined by the optimization algorithm and the following were determined:

1. FT/T - lower bound estimation of the number of employees,
2. m/NO - the ratio of the number of positions to the number of employees.

The optimization algorithm has been implemented in C++ in the Visual Studio 2010 environment and launched on a computer with an Intel i7 2.4 GHz processor working under the Windows 8.1 operating system. The operation time of the optimization algorithm for each product did not exceed several dozen seconds. The analysis of the experimental results collected in Table 3 shows that in all 20 cases, the algorithm has determined the optimal number of operators equal to the lower bounds estimation. It should be noted that determining the lower

Table 3. Results of experimental research

Product	m	FT	T	NO	FT/T	m/NO
1	9	10.197	2.39	5	5	1.8
2	10	11.049	2.39	5	5	2
3	12	12.579	2.989	5	5	2.4
4	12	12.579	2.989	5	5	2.4
5	12	13.431	2.989	5	5	2.4
6	5	19.129	10.6	2	2	2.5
7	5	19.563	10.6	2	2	2.5
8	5	13.258	9.204	2	2	2.5
9	6	7.452	2.232	4	4	1.5
10	7	13.319	4.061	4	4	1.7
11	8	14.399	4.061	4	4	2
12	8	17.498	7.416	3	3	2.6
13	9	18.212	7.416	3	3	3
14	6	13.002	4.41	3	3	2
15	5	18.09	12.766	2	2	2.5
16	3	5.202	2.487	3	3	1
17	3	4.044	1.825	3	3	1
18	3	4.044	1.825	3	3	1
19	3	4.405	2.186	3	3	1
20	3	12.321	5.187	3	3	1

bound of the number of employees is trivial, but determining the allocation of operators to machines and technological activities and determining a schedule for their implementation for most products is very difficult and requires the use of the proposed algorithm.

The values of the m/NO coefficient range from 1 to 2.6. The value means that for certain products the number of employees is equal to the number of machines, so each employee only serves one machine. In the case of the highest value of this coefficient, the average employee supports as many as 2.6 machines, i.e. 3 employees operate 8 machines in the production cell with full efficiency.

6 Summary

The work considers the problem of determining the smallest number of operators in a multi-station manufacturing cell. The original concept of the so-called a cyclic core simplifying the sequence of servicing many positions by employees. The method of constructing the order of performing all technological activities and the schedule of their implementation has been proposed.

An optimization algorithm based on the method of an all-round review of all cyclic cores has been proposed. The algorithm eliminates the generation of cyclic cores for which the schedule would not guarantee operation of the cell with a full productivity. Research has been carried out on 20 products manufactured in a company that makes car seats. For all products, a minimum number of employees was set equal to the lower estimate and an acceptable work schedule.

The method of constraints modeling proposed in the work and the optimization algorithm can be used to design manufacturing cells and determine the number of employees, assign them to machines, set a cyclical schedule of their implementation for new production cells and improve the existing ones. The proposed exact algorithm is proposed, therefore the number of employees is determined optimally.

Acknowledgement. The paper was partially supported by the National Science Centre of Poland, grant OPUS no. DEC 2017/25/B/ST7/02181.

References

1. Bożejko, W., Gnatowski, A., Pempera, J., Wodecki, M.: Parallel tabu search for the cyclic job shop scheduling problem. Comput. Ind. Eng. **113**, 512–524 (2017)
2. Bożejko, W., Pempera, J., Wodecki, M.: A fine-grained parallel algorithm for the cyclic flexible job shop problem. Arch. Control Sci. **27**, 169–181 (2017)
3. Bożejko, W., Gnatowski, A., Klempous, R., Affenzeller, M., Beham, A.: Cyclic scheduling of a robotic cell. In: Proceedings of 7th IEEE International Conference on Cognitive Infocommunications CogInfoCom 2016, 16–18 October, Wrocław, Poland, pp. 379–384 (2016)
4. Bożejko, W., Uchroński, M., Wodecki, M.: Parallel metaheuristics for the cyclic flow shop scheduling problem. Comput. Ind. Eng. **95**, 156–163 (2016)
5. Bożejko, W., Uchroński, M., Wodecki, M.: Block approach to the cyclic flow shop scheduling. Comput. Ind. Eng. **81**, 158–166 (2015)
6. Che, A., Chu, C.: Multi-degree cyclic scheduling of a no-wait robotic cell with multiple robots. Eur. J. Oper. Res. **199**(1), 77–88 (2009)
7. Drobouchevitch, I.G., Sethi, S.P., Sriskandarajah, C.: Scheduling dual gripper robotic cell: one-unit cycles. Eur. J. Oper. Res. **171**(2), 598–631 (2006)
8. Grabowski, J., Pempera, J.: The permutation flow shop problem with blocking. A tabu search approach. Omega **35**, 302–311 (2007)
9. Levner, E., Kats, V., Levit, V.E.: An improved algorithm for cyclic flowshop scheduling in a robotic cell. Eur. J. Oper. Res. **97**(3), 500–508 (1997)
10. Pempera, J., Smutnicki, C.: Open shop cyclic scheduling. Eur. J. Oper. Res. **269**, 773–781 (2018)
11. Ronconi, D.P.: A branch-and-bound algorithm to minimize the makespan in a flowshop with blocking. Ann. Oper. Res. **138**, 53–65 (2005)
12. Ronconi, D.P.: A note on constructive heuristics for the flowshop problem with blocking. Int. J. Prod. Econ. **87**, 39–48 (2004)