Applied Intelligence (2022) 52:3445–3464
https://doi.org/10.1007/s10489-021-02594-x

Proactive and reactive approach to employee competence configuration problem in planning and scheduling processes

Jarosław Wikarek[1] · Paweł Sitek[1]

Accepted: 4 June 2021 / Published online: 7 July 2021

Abstract
At the time of commonplace automation, robotization and the rapid development of IT, high qualifications of employees have become the critical element of every industry system. This follows from their limited availability, frequently high costs of procurement and possible employee absenteeism. Moreover, the concept of Industry 4.0 will transform current industry employees into knowledge employees. This is due to the fact that hard and routine tasks will be executed by robots and computers. This constitutes change in the required employee competences. Unfortunately, the aspect of management and configuration of employee competences is often overlooked in industrial practice. In response to the existing problem, the article puts forward the original model of employee competence configuration which is a basis for responses to numerous questions of managers of production processes, both general ones, e.g., *Do we have a sufficient set of competences to execute a production schedule?* as well as detailed ones, e.g., *Which and how many competences are missing?* etc. An important novelty of the presented model is the possibility of its application with both proactive and reactive questions. Due to the discrete and combinatorial nature of the problem under consideration, the use of mathematical programming methods was limited only to small data instances. Therefore, a proprietary dedicated genetic algorithm was proposed to solve this problem, which turned out to be extremely effective. The use of this genetic algorithm has enabled finding a solution depending on the instance data up to 70 times faster than by use of the mathematical programming.

Keywords Proactive and reactive approach · Scheduling · Resource allocation · Genetic algorithms · Constraint logic programming · Mathematical programming

1 Introduction

Problems with planning, scheduling, and the control and allocation of resources are the primary issues that should be taken into account and solved during the production control [15, 16, 20]. They occur in production with a job-shop, flow-shop, open-shop and project-type organization [11]. They refer to automated and flexible manufacturing systems as well as to smart manufacturing [23]. In diverse forms and scopes, they are also present in logistics, distribution, transport, supply chain, planning of classes at universities, etc. Literature devoted to this issue is very extensive (Section 2). A number of tools and IT systems have been developed to help and optimize decisions in the area of planning, scheduling and allocation of usually constrained resources in production control. The most interesting systems of this class of APS (Advances Planning Scheduling) systems include: Asprova APS, DSX, Preactor APS, etc. (https://www.g2.com/categories/advanced-planning-and-scheduling-aps, Accessed April 042021) as well as other dedicated systems and algorithms [23]. The result of the operation of such systems is a plan/schedule of task completion over time, allocation of tasks to specific machines/work stations/CNC machines, vehicles, etc. In more advanced systems, there is also a possibility of allocating tasks to constrained additional resources such as, e.g., tools, employees, software, etc. and a possibility of a dynamic change of schedule, reception of an answer to the *what-if* question in case of changes in task performance or machine unavailability.

✉ Jarosław Wikarek
sitek@tu.kielce.pl

Paweł Sitek
j.wikarek@tu.kielce.pl

[1] Kielce University of Technology, Kielce, Poland

Table 1 Employee competence

	e_1	e_2	e_3	e_4	e_5	e_6
i_1	1	0	0	1	1	0
i_2	1	0	0	1	1	0
i_3	0	0	1	0	1	1
i_4	0	0	1	0	0	1
i_5	0	1	0	1	1	0
i_6	1	1	0	1	0	0
i_7	1	1	0	0	1	0
i_8	0	1	0	1	1	0
i_9	0	0	1	0	1	1
i_{10}	0	0	1	0	1	0

Table 2 Competence required for performance of individual tasks from orders $o_1..o_4$

	e_1	e_2	e_3	e_4	e_5	e_6
k_1	1	0	0	0	1	0
k_2	0	0	1	0	0	1
k_3	0	1	0	1	0	0
k_4	1	0	0	0	1	0
k_5	0	1	0	0	1	0
k_6	1	0	0	0	1	0
k_7	0	1	0	1	0	0
k_8	0	0	1	0	0	1
k_9	0	0	1	0	1	0
k_{10}	0	1	0	0	1	0
k_{11}	1	0	0	1	0	0
k_{12}	0	0	1	0	1	0
k_{13}	1	0	0	1	0	0
k_{14}	0	1	0	0	1	0
k_{15}	0	0	1	0	1	0

In the modern production processes, which are characterized by a significant degree of automation, robotization and saturation with state-of-the-art IT technologies, especially in smart manufacturing, the problem of constrained production resources is no longer as crucial as it used to be for the production systems of past generations. The critical resources in modern production systems are employees who have high and specialist competences, qualifications, and experience, etc. This follows from the fact that very often they cannot be replaced with technological solutions. An additional problem related to this type of resource (i.e., employee competence) is its slight availability on the market, high costs and risk of employee absence during completion of the industrial process.

The paper proposes a model of configuration, selection and allocation of employee competence for process performed according to a pre-defined plan/schedule. Thus, the proposed model may be used to support decisions within the scope of selection, configuration and supplementation of the set of available employee competences in the production/distribution/organizational process, both in a proactive and reactive mode (Section 3). In particular, it allows for finding answers to a number of questions, both general and detailed, such as: *Do we have a sufficient set of competences to execute a pre-set plan/production schedule? Which and how many competences are missing? Which and how many competences will be missing if specific employees are absent?* etc. Moreover, the proposed model can be used to support decisions both in a proactive and reactive way.

The main contributions of this work are: (a) the original mathematical model of the problem used in the proactive and reactive approach, (b) the AMPL implementation model, (c) the structure of data for the modeled problem in Graph NoSQL Database, and above all (d) an original method of solving the modeled problem in the form of a dedicated genetic algorithm. The modifications concerned the introduction to the algorithm of the presolving procedure and the original method of representing the modeled problem.

The rest of the article is organized in the following way. Section 2 provides a literature overview of the problem of staff allocation in scheduling problems. Section 3 provides a comprehensive description of the modeled problem, including illustrative example, mathematical model, scope of decision support, etc. Implementation methods and computational examples are given in Section 4. The final section is a summary and conclusions.

2 Literature review

The issue of allocating employees to tasks pending execution, project stages, work positions, etc. is most often formulated in literature as employee timetabling or staff scheduling [10, 13].

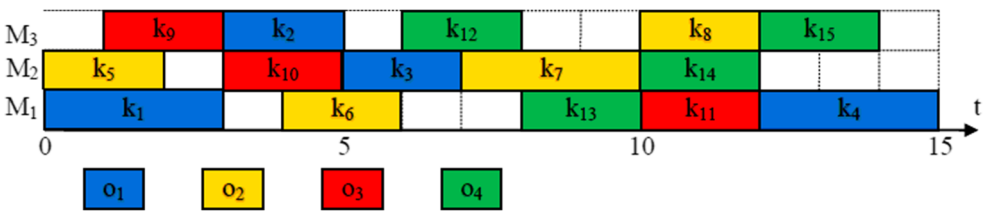

Fig. 1 Task performance schedule (every order is marked with a different color)

Table 3 Possible allocations of employees to tasks

	o_1				o_2				o_3			o_4			
	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}
i_1	1	0	0	1	0	1	0	0	0	0	1	0	1	0	0
i_2	1	0	0	1	0	1	0	0	0	0	1	0	1	0	0
i_3	0	1	0	0	0	0	0	1	1	0	0	1	0	0	1
i_4	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
i_5	0	0	1	0	1	0	1	0	0	1	0	0	0	1	0
i_6	0	0	1	0	0	0	1	0	0	0	1	0	1	0	0
i_7	1	0	0	1	1	1	0	0	0	1	0	0	0	1	0
i_8	0	0	1	0	1	0	1	0	0	1	0	0	0	1	0
i_9	0	1	0	0	0	0	0	1	1	0	0	1	0	0	1
i_{10}	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1

Such problems refer to finding work timetables for an organization's staff which allow for performance of a specific set of orders, services, projects, etc. There are a number of varieties of this problem such as university timetabling problem, nurse scheduling problem (NSP), crew scheduling problem, etc. The examined problems may differ by objective functions, constraints, etc., as well as by computational complexity. In practice, on account of the discrete and combinatorial character of the above-listed issues, these are most frequently NP-hard problems.

In industrial practice, the problem of allocation of employees is usually analyzed in the context of classic problems of planning and scheduling, such as machine scheduling problem, also known as operation problem [1, 6, 24]. The machine scheduling problem is common in the machine, electronic, IT and automotive industry, etc.

Both of the optimization problems listed above, namely employee allocation and machine scheduling, are most often analyzed together and consist in the allocation of employees to machines/work positions with the aim of performing a given production schedule. The literature is dominated by an approach consisting in sequential solving of both problems, i.e., first the employee/personnel allocation problem is solved and subsequently, relying on the received solution, the machine scheduling problem is tackled. Such approach usually results in finding sub-optimal solutions for the entire issue (an optimum solution for the first problem does not guarantee an optimum solution for the second one). There are also integrative approaches where both problems are solved jointly. Such approaches were proposed, for example, in [33, 7, 8]), where employees were treated as an additional resource indispensable to service machines and handle tasks, and where the problem of scheduling with additional constraints was solved.

Both accurate and approximate methods are used to solve integrated problems. The most promising accurate methods include hybrid approaches integrating mathematical programming methods and constraint programming [3, 26, 27]. As far as approximate methods are concerned, these are dedicated heuristics and meta-heuristics [2, 24]. A significant drawback of integrative approaches is their inability to handle the increase in the dimension of the modeled problem, which results in a higher number of decision variables and constraints, and this in turn significantly increases the time of solving the modeled problem.

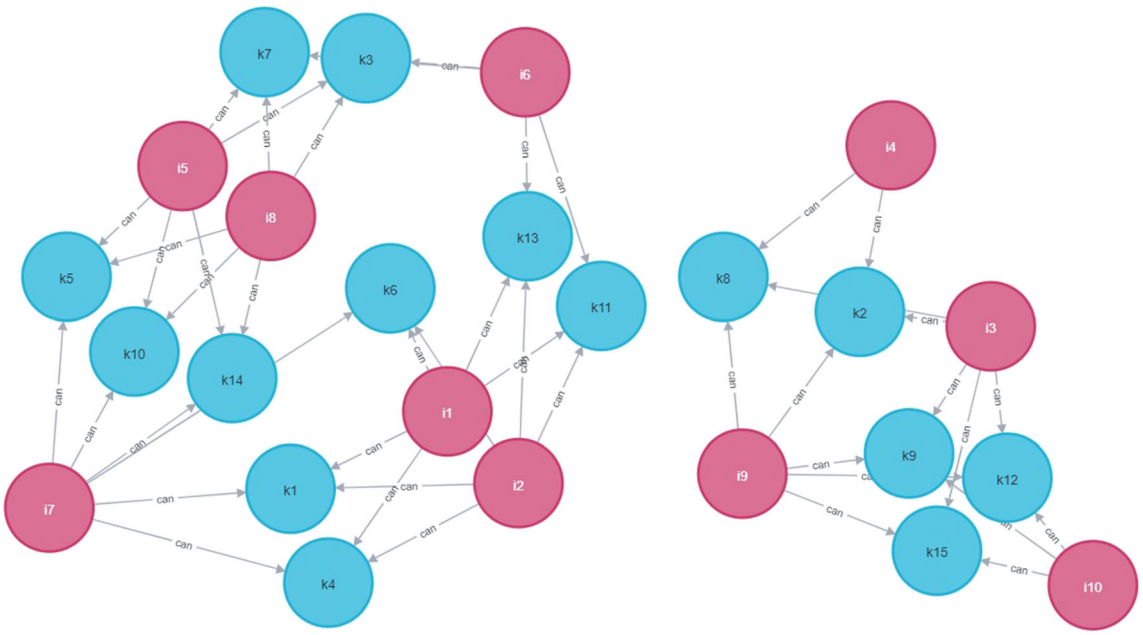

Fig. 2 Employee competence required for performance of a set of tasks for the illustrative example *Example_1*

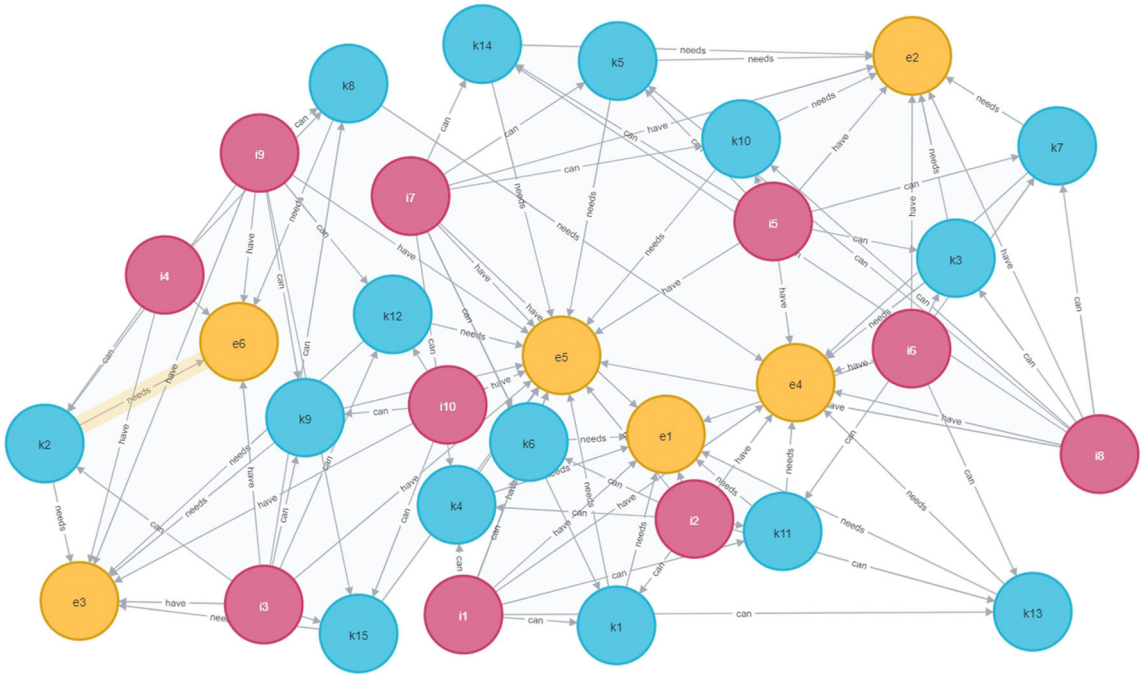

Fig. 3 Potential allocations of employees to tasks on account of required competence for the illustrative example *Example_1*

Very interesting approaches have been proposed in relation to Industry 4.0 in [19, 21], which confirm the transformation of employee competences from performing simple tasks to knowledge-based activities including technical, transversal and contextual skills.

The approach proposed by the authors, as opposed to the integrative and sequential approach, allows solving a problem with smaller dimensions. This is due to the fact that it is not necessary to solve the scheduling problem, but only to determine the required employee competences (whether the company has them, which are missing, etc.) necessary to perform the tasks according to given schedule. This is justified, because in practice the contracting authority often imposes a predetermined schedule for the implementation of projects, production or transport tasks. Additionally, in the proposed approach we examine employee competence (which can be

Table 4 Questions about the issue of employee competence configuration

Q	Question	Mode
Q_1	Is the set of employee competences held by us sufficient to perform tasks according to the pre-defined schedule?	R
Q_2	What is the minimum set of competences to perform a specific set of tasks according to the pre-defined schedule?	R
Q_3	Which competences and in which number should they be supplemented in order to perform the specified set of tasks according to the pre-defined schedule?	R
Q_4	Is the held set of competences (configuration) sufficient to implement a specific set of tasks independently from the fact of which case of unavailability of employees $u_j \in U$ takes place? (u_j absence of random employee, absence of two random employees, etc., absence of several selected employees, etc.)	R/P
Q_5	Which set of competences (configuration) guarantees implementation of a specific set of tasks irrespective of the fact of which case of unavailability of employees $u_j \in U$ takes place?	P
Q_6	Which competences and in which number should be supplemented to guarantee execution of a specific set of tasks irrespective of the fact of which case of unavailability of employees $u_j \in U$ takes place?	P
Q_7	Which set of competences (configuration) guarantees execution of a specific set of tasks with the p probability if a case of unavailability of employees $u_j \in U$ takes place?	P
Q_8	Which competences and in which number must they be supplemented in order to guarantee execution of a specific set of tasks with the p probability if a case of unavailability of employees $u_j \in U$ takes place?	P

P, proactive; *R*, reactive

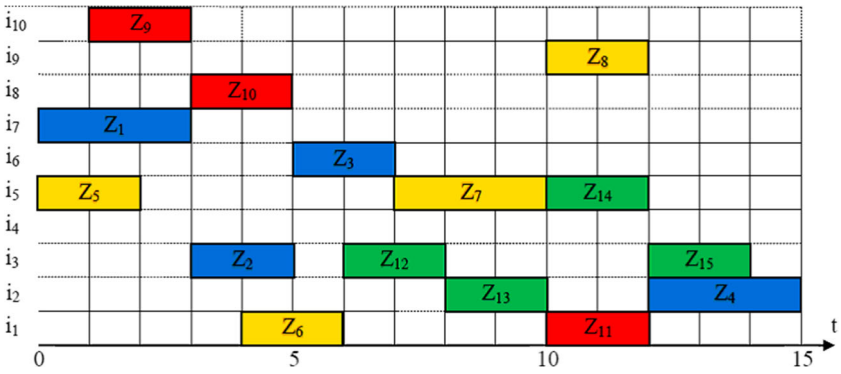

Fig. 4 Sample allocation of employees to tasks for *Example_1* generated by the operator, compliant with competences held and a pre-defined schedule (Fig. 1)

multidimensional and multilevel) and not simple permissible or impermissible allocations of employees to specific tasks. The presented approach also makes it possible to indicate which competences and which employees should be supplemented with them.

The proposed implementation methods turned out to be more effective and flexible than the methods of mathematical programming usually applied to such problems [5, 9, 18]. This concerned both the hybrid approach (Section 4.1) and the dedicated genetic algorithm (Section 4.2) developed by the authors. The particular effectiveness of this genetic algorithm resulted from the application of a dedicated presolving procedure. In the available literature on the application of evolutionary methods to solve discrete combinatorial problems [17, 31, 32], this is a unique solution.

3 Problem description

The problem of configuring employee competence in the industrial or organizational system may be formulated as follows: a specific set of tasks (projects), *K*, is to be performed. In

Table 5 Indices, parameters, decision variables

Symbol	Description
Indices	
I	Set of employees
K	Set of tasks
U	Set of unavailability of employees
i	Employee index ($i \in I$)
k	Task index ($k \in K$)
u	Index of unavailability of employees ($u \in U$)
Parameters	
$F_{u,i}$	If in unavailability state u employee i is available $F_{u,i} = 1$ otherwise $F_{u,i} = 0$
$G_{i,k}$	If the employee i has competence for execution of the task k, then $G_{i,k} = 1$ otherwise $G_{i,k} = 0$
$Go_{i,k}$	If the employee i may have competence for execution of the task k, then $Go_{i,k} = 1$ otherwise $Go_{i,k} = 0$
$Gk_{i,k}$	Cost of acquisition of a competence by the employee i for execution of the task k
$R_{k1,k2}$	If the task $k1$ is executed at the same time as $k2$ than $R_{k1,k2} = 1$, otherwise $R_{k1,k2} = 0$
St	Very large constant
Decision variables	
$X_{u,i,k}$	If during unavailability u ($u \in U$) the employee i executes the task k, then $X_{u,i,k} = 1$ otherwise $X_{u,i,k} = 0$
$Gx_{i,k}$	If execution of a set of tasks requires the employee i to accomplish competence for execution of the task k, then $Gx_{i,k} = 1$ otherwise $Gx_{i,k} = 0$
Wx_u	If all tasks are executed during unavailability u ($u \in U$), then $Wx_u = 0$ otherwise $Wx_u = 1$
$Y_{u,k}$	If the task k is not executed during unavailability u_j ($u_j \in U$), then $Y_{u,k} = 1$ otherwise $Y_{u,k} = 0$
Determined value	
Cost_Chang	Cost of acquiring competences the competencies to guarantee execution of tasks in every case
Control parameter	
L	Probability of execution (0: certainty 1: absence of one results in failure, 2: absence of two results in failure, etc.)

Table 6 Description of model constraint functions

Constraint	Description
(1)	Performance of all tasks
(2)	Change of only permitted competences
(3)	If the employee performs a task, he/she must hold adequate competence for it
(4)	Tasks are only performed by available employees
(5)	Determination of specific employees whose absence causes non-performance of tasks
(6)	Determination of probability
(7)	Binarity
(8)	Cost of accomplishing specific types of competence (different forms depend on question)
(9)	Objective function (Q2, Q3, Q5)
(10)	An employee does not execute two tasks at the same time

order to perform a k task, $k \in K$ one of the employees i from the I set has to be allocated to it ($i \in I$). For the k task to be performed by the i employee, the employee must have adequate competence. Various types of constraints may be imposed on tasks $k \in K$ and employees $i \in I$, specific for a given case (these constraints do not affect the methodology of solving the basic problem). The most important constraints of the modeled problem include constraints specifying:

- Work time limit of employee $i \in I$.
- Task performance schedule.
- Sequence of task performance, which usually follows from an imposed schedule, even though there may be additional determinants.
- A situation in which not every employee $i \in I$ can perform every task $k \in K$.
- A situation in which every employee $i \in I$ may have a specific number of competences (limit).
- Time for performance of a set of tasks resulting from a schedule or additional determinants.

The main question to which the answer is sought in a problem formulated in this manner is as follows: *Is the set of employee competences held by us sufficient to execute the tasks according to the pre-defined schedule?* In order to illustrate the problem of employee competence configuration, an illustrative example is given-*Example 1*.

3.1 Illustrative example

Example 1 A company is analyzed that has a production system with a job-shop type organization which consists of three different machines ($M_1 ... M_3$). The company employs 10 employees to service them ($i_1..i_{10}$). Employees may have six different types of competences ($e_1..e_6$). Table 1 presents the competence held by every employee (1 means that the employee has the given competence).

Four orders are to be performed ($o_1..o_4$). Order o_1 order consists of four tasks: k_1, k_2, k_3, k_4, order o_2 consists of four tasks k_5, k_6, k_7, k_8, order o_3 consists of three tasks k_9, k_{10}, k_{11} whereas order o_4 consists of four tasks k_{12}, k_{13}, k_{14}, k_{15}. In total, there are 15 tasks k to be performed. A task performance schedule is given and thus tasks on machines M_1, M_2 and M_3, which is shown in Fig. 1.

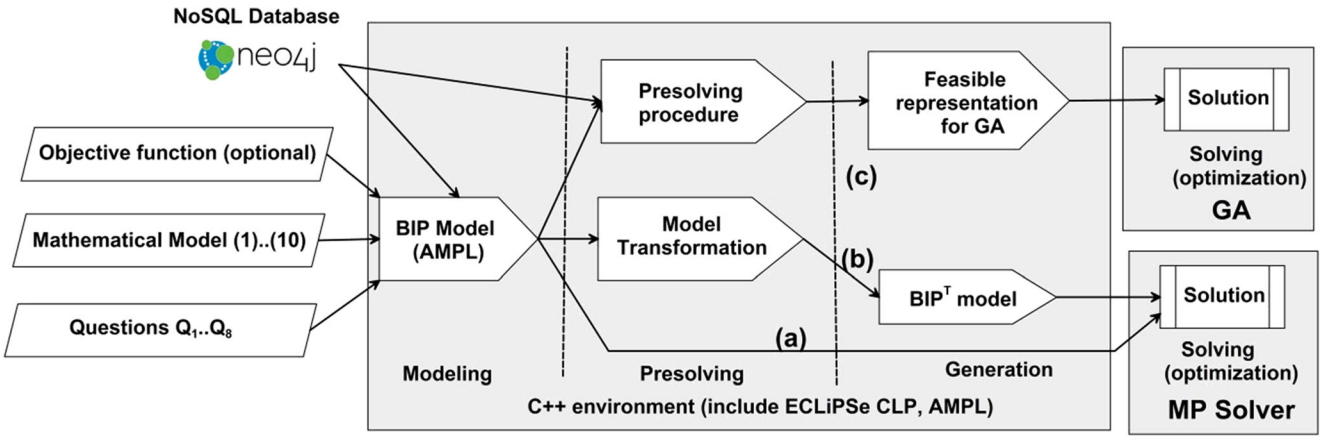

Fig. 5 Modeling and problem solving using an integrated framework

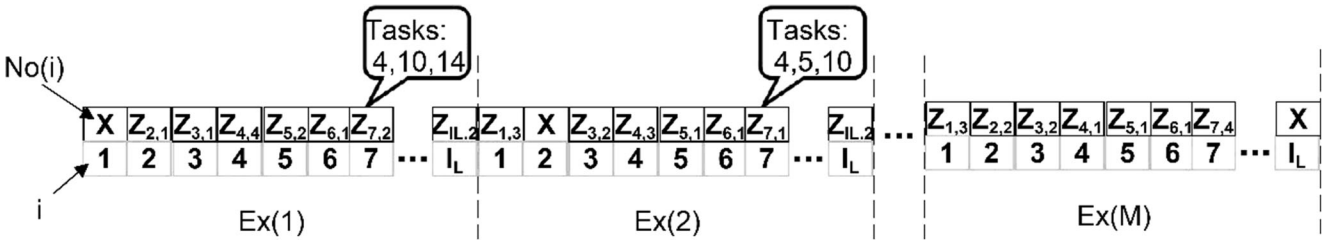

Fig. 6 The method of coding the problem (X-employee inaccessibility, Ex(m) - schedule execution method, I_L-last employee number)

Competence that is required for performance of individual tasks for an illustrative example is presented in Table 2 (1 specify the required competences to perform the task).

Attention: If, for example, the e_1 and e_5 competence is required for the k_1 task (in Table 2 marked with 1), this means that the allocated employee must have both types of such competence.

By combining the data from Tables 1 and 2, information is procured as to which employee can perform which task. Table 3 shows the possible allocation of employees to specific tasks as part of individual orders. Figure 2 presents all possible allocations of employees to tasks, whereas Fig. 3 shows both employees' competences, as well as possible allocations for performance of individual tasks. Figures are presented in the form of graphs generated from the Neo4j database, where data visible in Tables 1, 2 and 3 is presented (https://neo4j.com/).

Figure 4 presents a Gantt chart which shows one of the possible allocations of employees to tasks that guarantees performance of the set of tasks according to a pre-defined schedule (Fig. 1). The proposed example is also a basis for a positive answer to the question: *Do we have a sufficient set of competences to execute a set of tasks in line with the pre-defined schedule?* The allocation from Fig. 4 was generated by the operator based on data from Tables 1, 2 and 3. This required multiple adjustments, certain dexterity and the operator's experience, as well as time calculated in minutes. In a number of practical cases, such manual allocation does not guarantee success, whereas the number of iterations and changes increases quickly along with growth of dimensions of the analyzed problem. In such (manual) mode, it is hard to quickly find answers to other decision-making questions presented in Table 4. Therefore, a mathematical model was proposed for configuration of employee competences (Section 3.4) which allows for it.

Table 7 Combination of possible tasks for employee $i = 7$

No(i)	Employee(i)	Tasks
$Z_{7,1}$	7	4,5,10
$Z_{7,2}$	7	4,10,14
$Z_{7,3}$	7	1,4,6,14
$Z_{7,4}$	7	1,4,10,14
$Z_{7,5}$	7	4,5,6,14
$Z_{7,6}$	7	5,6,10,14

$\mathbf{Z_7} = \{Z_{7,1}. Z_{7,2}. Z_{7,3}. Z_{7,4} Z_{7,5}. Z_{7,6}\}$

3.2 Unavailability of employees

A significant aspect of the analyzed problem of employee competence configuration is the fact that the i ($i \in I$) employees may be unavailable due to various causes (e.g., sick leave, quarantine, planned holiday leave, training, etc.). Assume that, u_j denotes the case of unavailability of selected employees (e.g. $u_1 = \{i_3, i_7\}$ means that we are analyzing a case where the i_3, i_7 employees are unavailable, whereas if $u_2 = \{i_3\}$ it denotes a case where the i_3 employee is unavailable, where $i_3, i_7 \in I$, etc.). U denotes a set of all such analyzed (interesting to us) cases of employee unavailability $u_j \in U$. It is easy to calculate that if we have 10 employees and if we examine all possible absences of an individual employee, the U set will comprise 10 elements, if we want to account for all possible absences of two employees, the U set will comprise 45 or 55 elements if we also take individual absences into account, etc. In operating practice, a situation where the U set has to be taken into account with elements specifying absence of random two, three, four, etc. employees happens very rarely. Most frequently, there are absences of individual employees resulting from holiday leave, trainings or sudden events such as accidents, illness, etc. Nevertheless, for example, to ensure

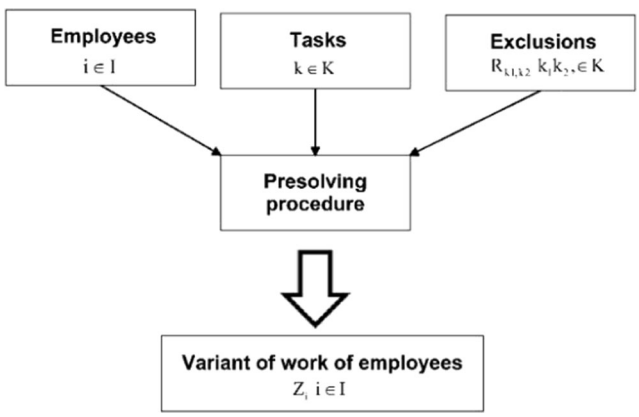

Fig. 7 The inputs and outputs of presolving procedure

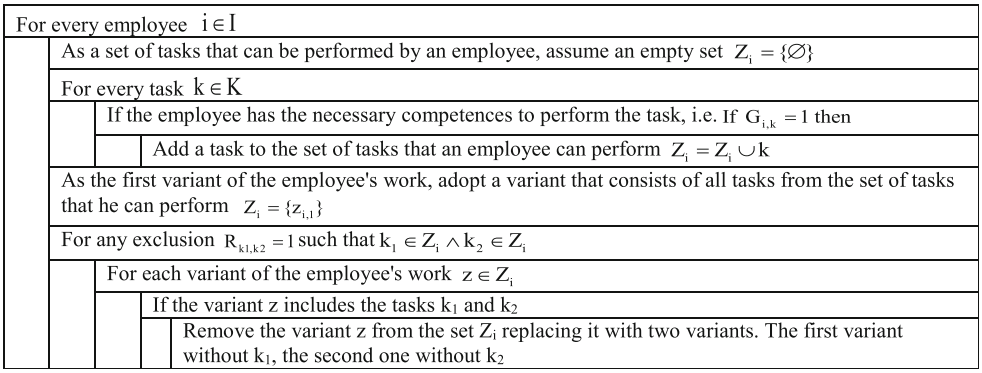

Fig. 8 Presolving procedure algorithm

robustness of the schedule of task performance to employee absenteeism (lack of specific competence), the *U* set has to account for all possible absences of employees, which results in a significant increase in the number of its elements. Given the limited number of available employees which results in limited availability of employee competence, the *p* probability of execution of a set of tasks with the occurrence of certain cases of employee unavailability was determined. Such probability was determined with the use of *Definition_1.*

Definition_1 The set of unavailability cases has the form of $U = \{u_1, u_2, \ldots, u_{LK}\}$, wherein *LK* is the number of all elements in the set *U*. The *L* parameter specifies the number of elements from the *U* set for which performance of a set of tasks will not be possible ($L = 0$ means that when any case of unavailability of u_j takes place, tasks will be performed; $L = 3$, means that when three cases of unavailability of u_j occur (e.g., if a situation u_3 or u_7 or u_{11}, etc. occurs), tasks will not be executed. On the other hand, if another case of unavailability of u_j occurs, tasks will be performed. The *L* parameter may be converted to the *p* probability of task execution with the *L* unavailability of elements from the *U* set. The mode of calculation of the *p* probability is presented with the use of formula (f1).

$$p = \frac{LK - L}{LK} \cdot 100 \tag{f1}$$

3.3 Scope of decision support

In reference to the presented issue of employee competence configuration, numerous questions can be formulated to which the decision-maker has to find an answer in planning and scheduling processes. These are both general and detailed questions. Obviously, they refer to the aspect of employee competence. The most important of them are presented in Table 4.

Such presentation of the problem and formulation of constraints allows for using the proposed model of the problem (Section 3.4) to assist the decision-making process both in a proactive and a reactive mode. In a proactive mode, the model allows for selection of such competences of individual employees that guarantees execution of a pre-defined set of tasks according to a specified schedule, in spite of the absence of any employee, any two employees, etc. In such mode, it is possible to receive answers to questions $Q_5 - Q_8$ (Table 4). A reactive mode model may also be used if during the execution of tasks or at the moment of commencement of their execution, information is received about the absence of specific employees. In this mode, it is possible to receive answers to questions $Q_1 - Q_4$ (Table 4). Obviously, many of these questions may also

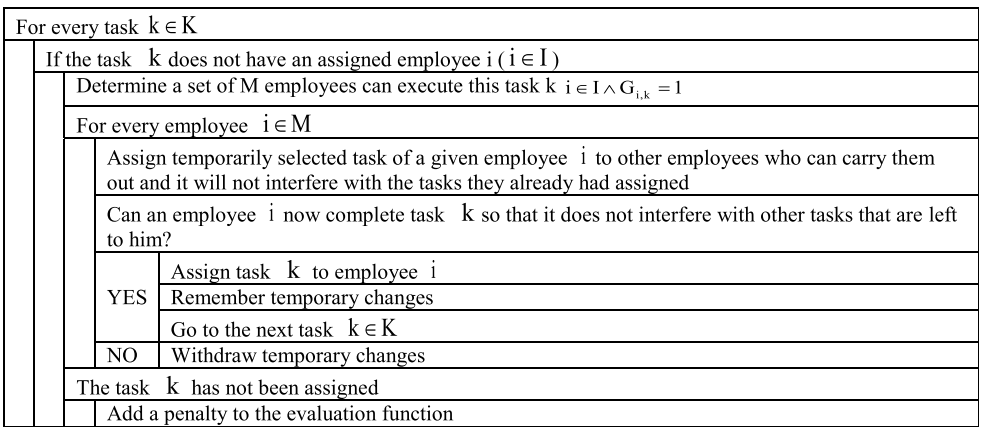

Fig. 9 Algorithm for repair function

be posed during processes of production, distribution, etc. with the pre-defined schedule at any T moment.

3.4 Mathematical model of the employee competence configuration problem

Based on the premises presented in Section 3, a formal model of the employee competence configuration problem (1)...(10) was proposed. It was formulated as BIP (Binary Integer Programming). Indices, parameters and decision variables of the models are presented in Table 5, whereas a description of the significance and the function of individual constraints is in Table 6. The formula of the model objective function depends on the $Q_1..Q_8$ question to which an answer is sought. For example, in order to receive an answer to questions Q_1 and Q_4, no objective function is required, but only fulfilment of the model's constraints. For question Q_2, the objective function (9) determines the minimum number of new competences that the employees must accomplish. For other questions, the objective function may have a slightly different form than (9).

$$\sum_{i\in I} X_{u,i,k} + Y_{u,k} = 1 \forall k\in K, u\in U \quad (1)$$

$$Gx_{i,k} \le Go_{i,k} + G_{i,k} \forall i\in I, k\in K \quad (2)$$

$$X_{u,i,k} \le Gx_{i,k} \forall i\in I, k\in K, u\in U \quad (3)$$

$$X_{u,i,k} \le F_{u,i} \forall i\in I, k\in K, u\in U \wedge i\in U \quad (4)$$

$$\sum_{k\in K} Y_{u,k} \le St\cdot Wx_u \forall u\in U \quad (5)$$

$$\sum_{u\in U} Wx_u \le L \quad (6)$$

$$\begin{aligned} X_{u,i,k} &\in \{0,1\} \forall i\in I, k\in K, u\in U \\ Gx_{i,k} &\in \{0,1\} \forall i\in I, k\in K \\ Y_{u,k} &\in \{0,1\} \forall u\in U, k\in K \\ Wx_u &\in \{0,1\} \forall u\in U \end{aligned} \quad (7)$$

$$\text{Cost\_Chang} = \sum_{i\in i}\sum_{k\in K} \left(Gk_{i,k}\cdot Gx_{i,k}\right) \quad (8)$$

$$\text{Min Cost\_Chang} \quad (9)$$

$$X_{u,i,k1} + X_{u,i,k2} \le 1 \forall i\in I, k1, k2\in K, u\in U \wedge R_{k1,k2} = 1 \quad (10)$$

In practice, the proposed model can be the basis for building a decision support system for managers in the area of production and/or distribution of any form of organization. Such a system may support the manager in making decisions regarding real situations, such as: (a) Whether to proceed with the execution of the order(s) based on the resources available?, (b) Should training be organized for employees in order to obtain specific competences?, (c) How to plan a budget related to supplementing employees' competences?, (d) How to organize a team of employees so that they can carry out orders in the event of their absence? etc.

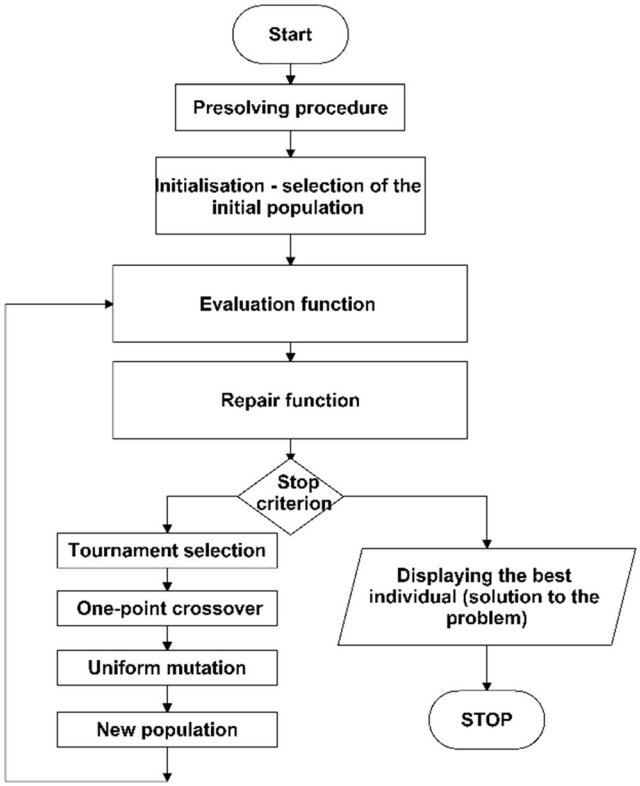

Fig. 10 Diagram of the dedicated genetic algorithm used

4 Methods and techniques

Due to the nature of the modeled problem (BIP-Binary Integer Programming), the MP (Mathematical Programming) environment seems natural to solve it. There are many MP solvers on the market such as: CPLEX, LINGO, S.C.IP, Gurobi etc. Unfortunately, due to the combinatorial nature of the problem and high computational complexity (NP-hard problem), their effectiveness is limited to problems of small size. Therefore, to model and solve the problem of competence configuration, a modified version of the proprietary hybrid approach was proposed [26, 27], which integrates methods and techniques of mathematical programming and constraint logic programming (CLP) in one integrated framework. For the presented problem, this framework has been supplemented with the dedicated genetic algorithm. The general framework diagram for this approach is shown in Fig. 5. The presented framework makes it possible to model and solve the problem of employee

Table 8 Scope of implementation of questions in individual experiment series

Series	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6
S_1	+	+	+	x	x	x
S_2	x	x	x	+	+	+
S_3	x	x	x	+	+	+

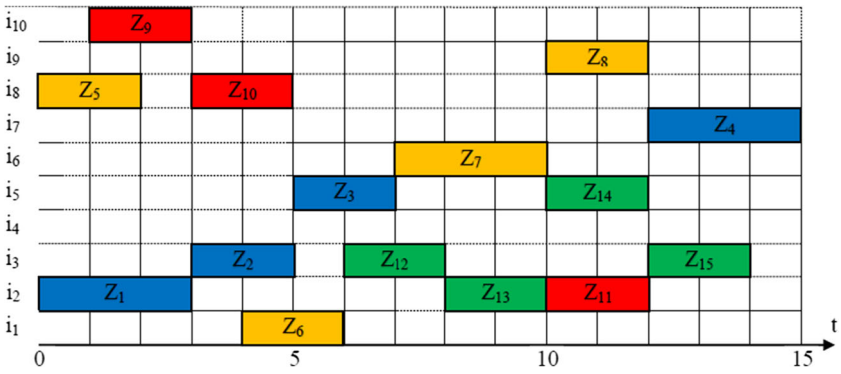

Fig. 11 Sample allocation of employees to tasks for the Q1 question

competence configuration using three alternative ways: (a) classical mathematical programming, (b) hybrid approach and (c) dedicated genetic algorithm. In the computational experiments, all three above methods were compared (Section 4.3). Section 4.1 describes the proprietary hybrid approach. The description of the dedicated genetic algorithm is presented in section 4.2.

For the implementation of the proposed integrated framework (Fig. 5), the following tools and environments were used: AMPL modelling language (https://ampl.com/), Neo4j (Graph NoSQL database) into which the data instances of the modeled problem (parameters, obtained outcomes, etc.) were uploaded, GUROBI mathematical and constraint programming solver (http://www.gurobi.com/, Accessed April 042021) and C++ for dedicated genetic algorithm.

4.1 Hybrid approach (b)

A proprietary hybrid approach was proposed as an alternative way of solving the modeled problem. [25, 28]. This approach uses presolving, which is the transformation of the modeled problem. The transformation is performed with constrained logic programming (CLP) methods and problem data instances. The obtained model after transformation is characterized by a smaller number of decision variables and constraints. Additionally, some of the constraints are simplified. This results in a significant reduction in the size of the solution space, which shortens the solution time. The model after transformation is solved using mathematical programming methods [26].

4.2 Dedicated genetic algorithm (c)

A dedicated genetic algorithm was proposed as a new way of solving the modeled problem. Compared to the classical genetic algorithm, it has been supplemented with the presolving procedure, specialized problem representation (Fig. 6), (Section 4.2.1), repair function (Section 4.2.2) .

4.2.1 Presolving procedure

The idea of presolving procedure is presented in Figs. 7, 8, while its location in the genetic algorithm is shown in Fig. 10. This procedure performs two functions simultaneously. On the one hand, it enables the generation of a specialized problem representation with the coding shown in Fig. 6 through direct constraint handling. On the other hand, by eliminating unacceptable allocation and exclusion, the size of the potential solutions is reduced. More precisely, to obtain chromosomes (individuals) with such coding, based on data and, all possible combinations of tasks were generated for each employee that he can perform during the implementation of a given schedule due to his competences. Each such combination of tasks was given a unique number/code *No(i)* and is the largest inseparable set of tasks to be performed due to a given schedule. For example, for employee $i = 7$, a list of combinations of tasks to be possible performed by him is presented in Table 7. The proposed representation and coding method (Fig. 6) ensures that each chromosome meets constraints (2), (3), (4), (5), i.e., the employee performs only those tasks for which he is competent. Practically, the rest of the constraints apart from (1) in the proposed coding method are also satisfied.

Each employee is assigned a No (i) code that specifies which tasks they can perform, and a set of all employees with assigned codes is a way to execute the Ex(m) schedule; a set of all such ways is one of the acceptable solutions to the configuration problem. The algorithm which is the basis of the presolving procedure is shown in Fig. 8.

4.2.2 Repair function

Unfortunately, the proposed coding (Fig.6) does not guarantee that the constraint (1) will be met. Therefore, a repair function has been developed to ensure that they are met. The function determines which tasks have not been allocated and then checks if there are employees who could perform them. If there are such employees, tasks are assigned to them. If this is not possible, a penalty for inadmissibility of the chromosome |(individual) is set. The detailed algorithm for repair

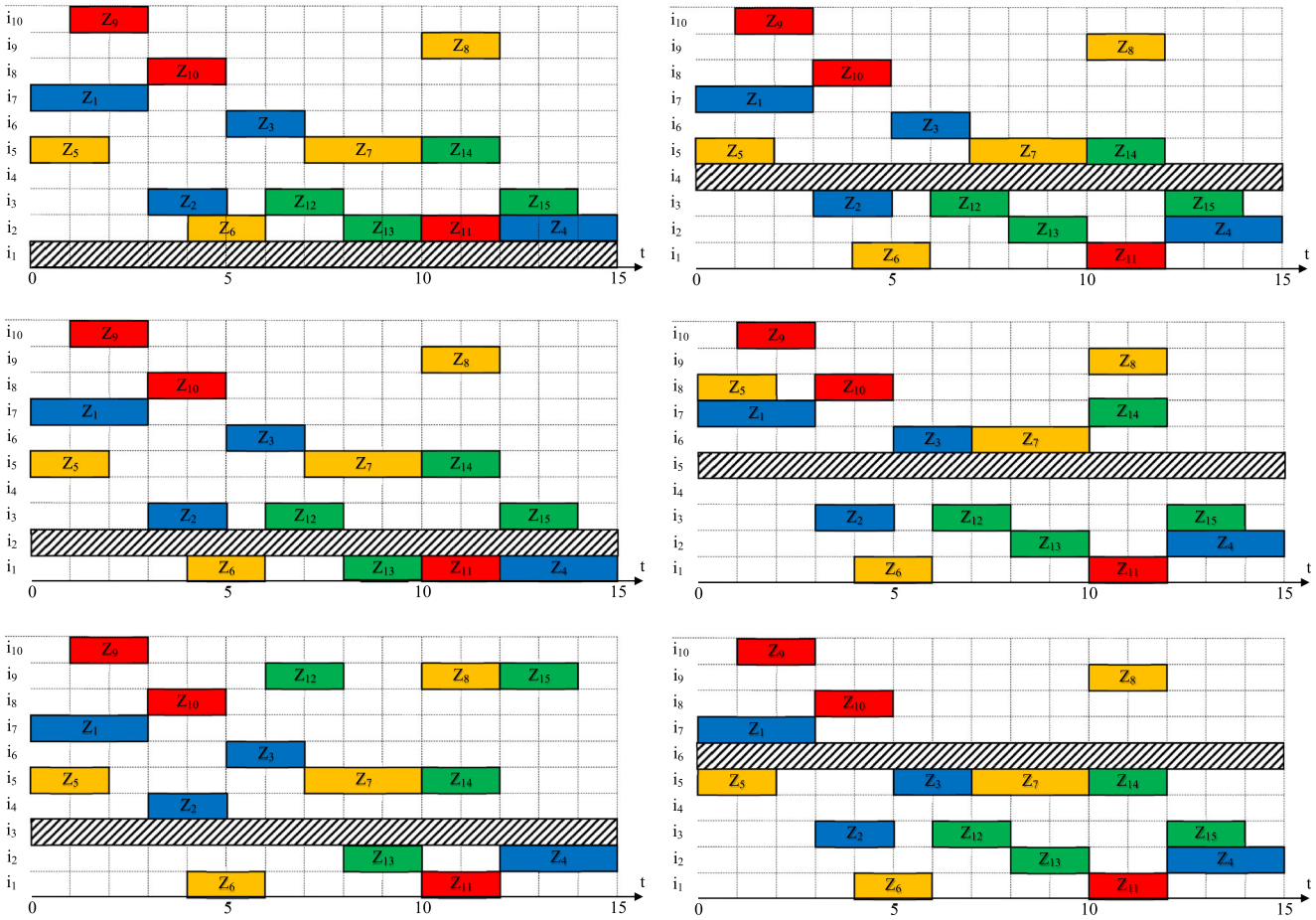

Fig. 12 Set of possible permissible allocations to tasks in case of the absence of a random employee (Q4 and Q5)

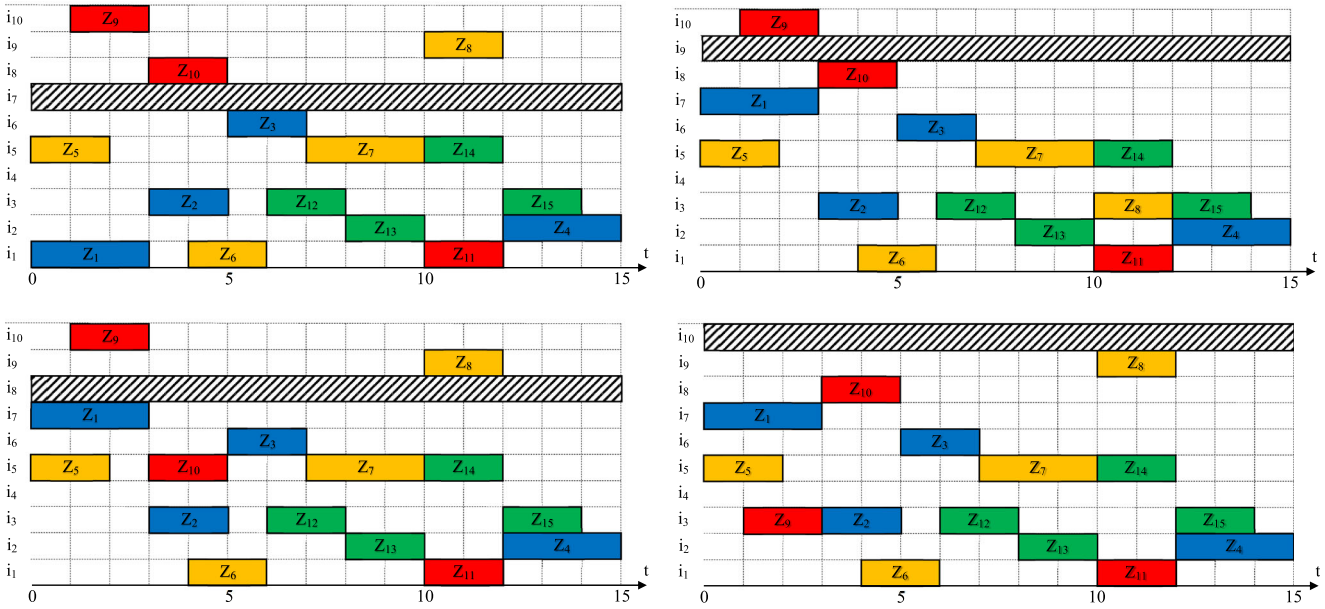

Fig. 12 (continued)

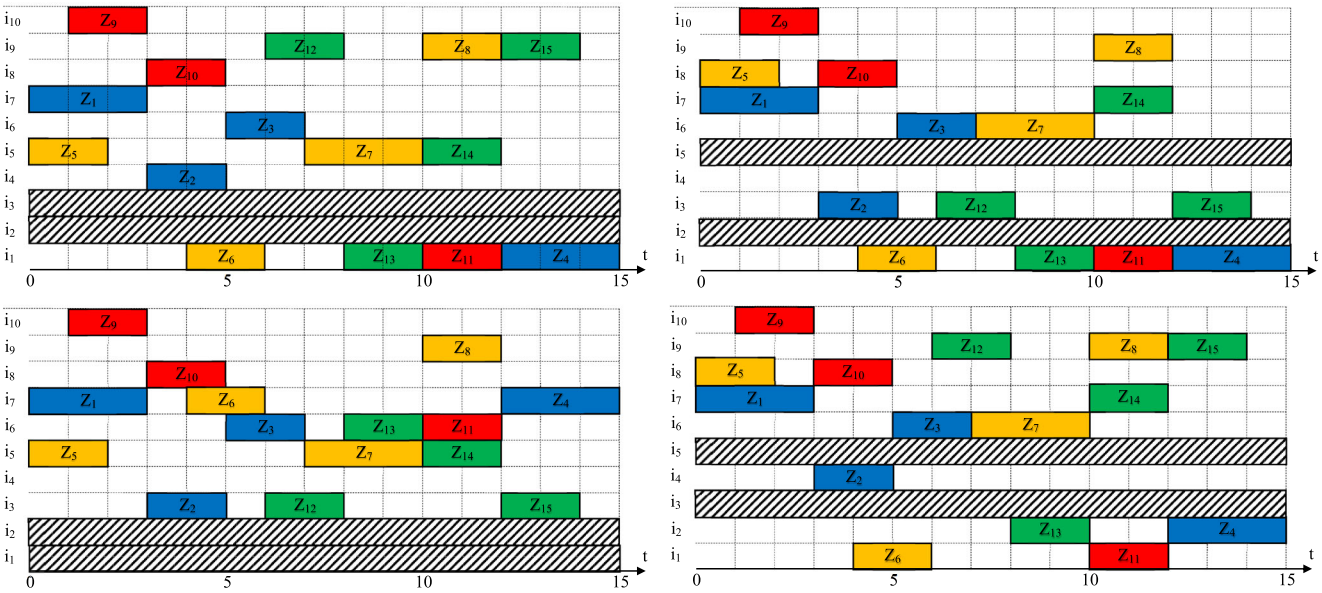

Fig. 13 Set of 4 sample permissible allocations to tasks in case of the absence of two random employees (Q4 and Q5)

function is presented in Fig. 9. The evaluation function was based on an assessment of each chromosome in terms of costs. Usually, these were the costs of the necessary changes or supplements to employee competences (*Cost_Chang*). In the case of impossibility to execute orders, a penalty was also added.

4.2.3 Genetic operators

Individuals are subject to genetic operations adapted to the presented problem, i.e., tournament selection, one-point crossover and uniform mutation. Tournament selection ensures proper selection pressure and prevents premature genetic algorithm convergence. On the other hand, crossover and mutation introduce appropriate diversity into the population. The genetic algorithm terminates when it reaches the number of iterations set by the program user or an additional stop criterion defined by the lack of improvement in the quality of the best individual in the population in the next 50 generations. For the proposed representation and genetic operators thus generated, the genetic algorithm is started according to the diagram in Fig. 10.

4.3 Computational examples

In order to verify the proposed mathematical model (section 3.4) and implementation, a number of calculation experiments were conducted. The experiments were carried out in two phases. At the first phase for the instance of data from the illustrative example *Example_1* (Section 3.1), experiments in three series were performed $S_1..S_3$. In the second phase of the experiments, the efficiency of the proposed model and implementation thereof was analyzed.

Series S_1 refers to a reactive approach, whereas series S_2 and S_3 refer to a much more interesting and more complex proactive approach. Table 8 presents details of experiments for individual series, i.e., which questions from Table 6 were taken into account in individual implementations for every series. In S_1 it was analyzed whether the held set of competences is sufficient to perform a set of tasks according to the pre-defined schedule (Fig. 1).

Implementation of the model allowed for finding all permissible allocations of employees to tasks which guarantees their execution according to the pre-defined schedule and gives a positive answer to the *Q1* question. One of such allocations in the form of a Gantt chart is presented in Fig. 11.

Table 9 Cases of unavailability of employees *(i)* that prevent schedule performance

u_j	Absent (i)	Absent (i)	Absent (i)
u_1	1	2	6
u_2	1	2	7
u_3	3	4	9
u_4	3	9	10
u_5	5	6	8

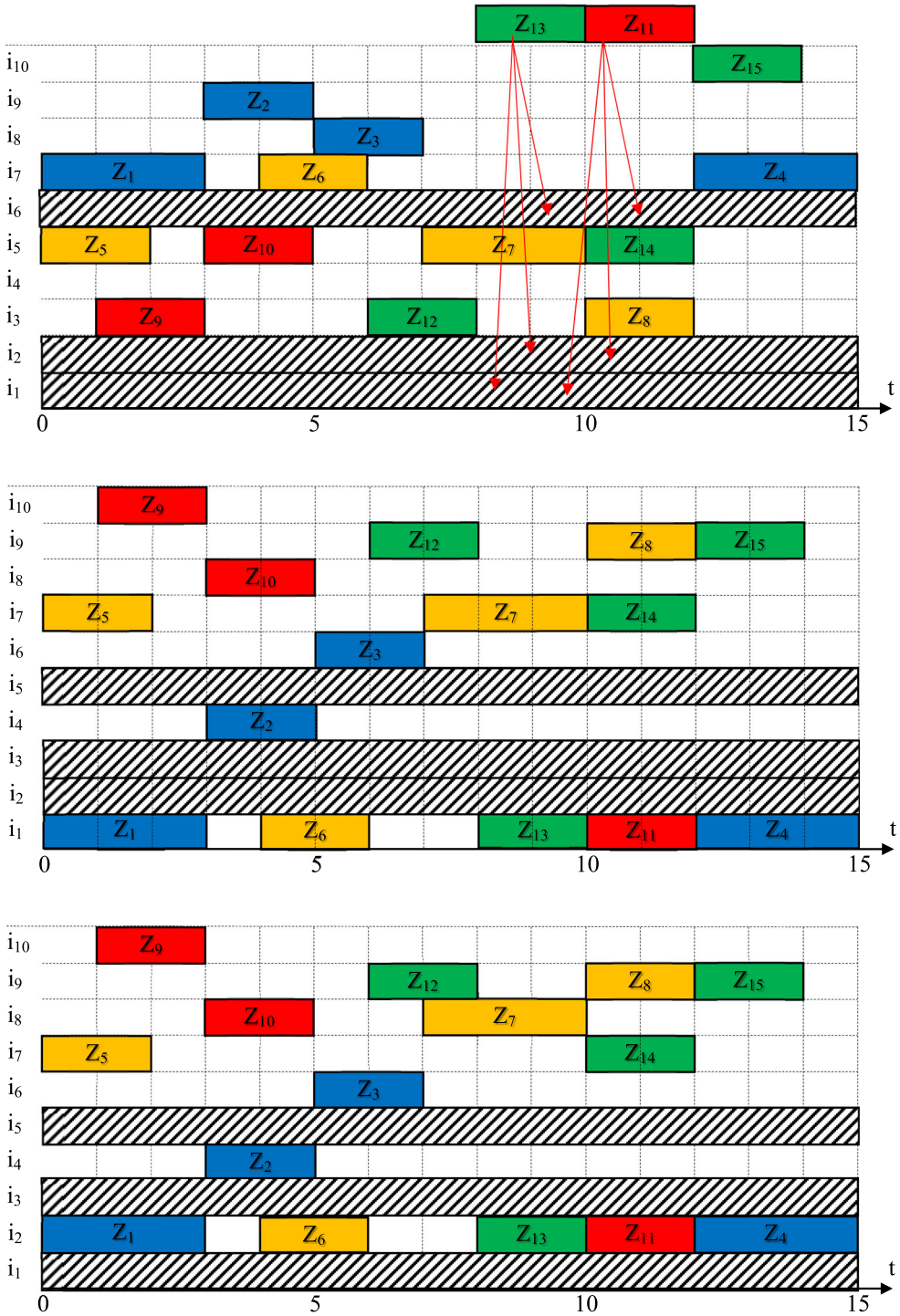

Fig. 14 Two permissible and one not permitted allocation of employees to tasks with respect to unavailability of three employees

In the second series of experiments (S_2), the impact of the absence of a random employee and two random employees was examined on the execution of tasks according to the pre-defined schedule (Fig. 1). In this series, implementation of the model taking into account various versions of the Q_4 question and the Q_5 and Q_6 questions was used. In both cases of absence, all possible permissible allocations of employees to tasks that guaranteed execution of the pre-defined schedule were found. In the first case, there were 10 such allocations

Table 10 New configuration of competence

	e_1	e_2	e_3	e_4	e_5	e_6
i_1	1	0	0	1	1	0
i_2	1	0	0	1	1	0
i_3	0	0	1	0	1	1
i_4	0	0	1	0	**1**	1
i_5	0	1	0	1	1	0
i_6	1	1	0	1	**1**	0
i_7	1	1	0	**1**	1	0
i_8	0	1	0	1	1	0
i_9	0	0	1	0	1	1
i_{10}	0	0	1	0	1	**1**

Table 11 New possible allocations of employees to tasks

	o_1				o_2				o_3			o_4			
	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}
i_1	1	0	0	1	0	1	0	0	0	0	1	0	1	0	0
i_2	1	0	0	1	0	1	0	0	0	0	1	0	1	0	0
i_3	0	1	0	0	0	0	0	1	1	0	0	1	0	0	1
i_4	0	1	0	0	0	0	0	1	**1**	0	0	**1**	0	0	**1**
i_5	0	0	1	0	1	0	1	0	0	1	0	0	0	1	0
i_6	**1**	0	1	**1**	**1**	**1**	1	0	0	**1**	1	0	1	**1**	0
i_7	1	0	**1**	1	1	1	**1**	0	0	1	**1**	0	**1**	1	0
i_8	0	0	1	0	1	0	1	0	0	1	0	0	0	1	0
i_9	0	1	0	0	0	0	0	1	1	0	0	1	0	0	1
i_{10}	0	**1**	0	0	0	0	0	**1**	1	0	0	1	0	0	1

(Fig. 12), whereas in the second 45. Figure 13 presents four permissible allocations of employees to tasks selected out of 45. For other data instances where the number of competences is insufficient to meet the schedule, implementation of the model from this series allows for finding an answer to questions Q_6.

The third series of experiments (S_3) refers to a situation where the impact of the absence of three random employees is examined. For the illustrative example *Example_1* it is possible to generate 120 of such allocations. The proposed implementation of the model included questions Q_4, Q_5, Q_6. It turned out that the answer to the Q_4 was negative if the $p = 1$ probability was taken into account. There were six cases of unavailability of $u_1..u_6$, which were presented in Table 9. For such cases it is impossible to generate the permissible allocation of employees to tasks, i.e., such allocation that would allow for performance of these tasks according to the pre-defined schedule (Fig. 1).

Obviously, one may venture saying that the answer to the Q_4 question is positive, yet with the probability of 0.95 ((120–6)/120 ≈ 0.95). Figure 14 presents three sample Gantt charts, the upper for the u_1 unavailability and the bottom two for permissible allocations.

In order to guarantee performance of a schedule with the $p = 1$ probability, it was necessary to adequately supplement competences. Table 10 presents a modified set of employee competences in relation to the configuration from Table 2. It was received by finding an answer to the Q_5 question. New competences that had to be acquired by adequate employees in order to guarantee performance of the schedule were marked in red and bold in the table (Q_6). Table 11 presents the new modified possible allocations of employees to tasks that allow for performance of the schedule in the absence of three random employees. Figure 15 presents allocation of employees to tasks in case of the unavailability u_1 which after supplementation of competences in line with Table 10 became a permissible allocation.

In the second stage of experiments, model implementation efficiency was examined. The experiments were carried out with the use of an integrated framework (Fig. 5), which

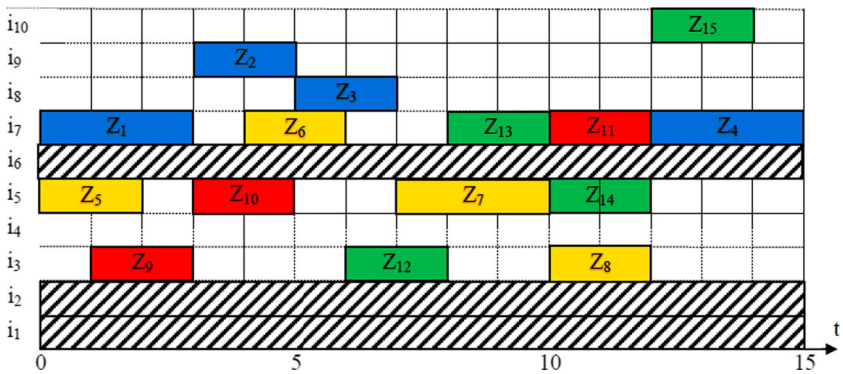

Fig. 15 Allocation of employees to tasks in the absence of three (*u1* from Table 9) after change of competence

Table 12 Results from reactive approach (Q4)

Input data					Result Q_4								
					MP Approach (a)				Hybrid Approach (b)			Dedicated Genetic Algorithm(c)	
In	E	I	K	U/W	p	v	T	F	V	T	F	T	F
In1	6	12	16	1/2	100	457	56 s	1	283	4 s	1	18	1
In2	6	12	16	1/3	80	457	51 s	0	283	5 s	0	4	0
In3	10	20	24	1/3	100	1081	345 s	2	497	6 s	2	21	2
In4	10	20	24	1/3	80	1081	323 s	0	497	7 s	0	6	0
In5	20	30	42	1/3	100	3121	651 s	2	1613	10 s	2	25	2
In6	20	30	42	1/3	80	3121	667 s	1	1613	11 s	1	28	1
In7	20	30	42	1/3	60	3121	649 s	0	1613	9 s	0	11	0
In8	20	30	42	1/3	40	3121	650 s	0	1613	9 s	0	10	0
In9	20	30	72	1/3	80	4345	845 s	1	2145	24 s	1	31	1
In10	20	40	72	1/3	40	5844	943 s	0	2843	34 s	0	21	0

In -number of data instance E: number of competences I: number of employees

v: number of decision variables K: number of tasks.

p: percentage of implementation certainty T: calculation time (in seconds)

F: number of required changes in competences (0 - the held set is sufficient for performance)

U/W: number of cases of unavailability/ number of unavailable employees

enables three methods of implementing the modeled problem: (a) mathematical programming, (b) hybrid approach and (c) dedicated genetic algorithm.

Two versions of the model were selected for the experiments. The first with the question Q4 (reactive example) and the second with the question Q5 (proactive

Table 13 Results from proactive approach (Q_5)

Input data					Result Q_5								
					MP Approach (a)				Hybrid Approach (b)			Dedicated Genetic Algorithm(c)	
In	E	I	K	U/W	p	v	T	F	V	T	F	T	F
In1	6	12	16	80	100	15,704	547	10	10,190	123	10	51	10
In2	6	12	16	80	80	15,704	576	6	10,190	134	6	53	6
In3	10	20	24	80	100	39,080	1356	14	18,450	345	14	97	14
In4	10	20	24	80	80	39,080	1434	12	18,450	323	12	99	12
In5	20	30	42	80	100	102,740	3456	24	54,111	456	24	174	24
In6	20	30	42	80	80	102,740	3567	20	54,111	467	20	175	20
In7	20	30	42	80	60	102,740	3245	16	54,111	434	16	170	16
In8	20	30	42	80	40	102,740	3432	12	54,111	481	12	181	12
In9	30	30	72	80	80	180,930	4354	12	92,325	654	10	197	10
In10	30	40	72	80	40	239,120	5432	6	122,323	698	6	204	6

In -number of data instance E: number of competences I: number of employees

v: number of decision variables K: number of tasks.

p: percentage of implementation certainty T: calculation time (in seconds)

F: number of required changes in competences (0 - the held set is sufficient for performance)

U/W: number of cases of unavailability/ number of unavailable employees

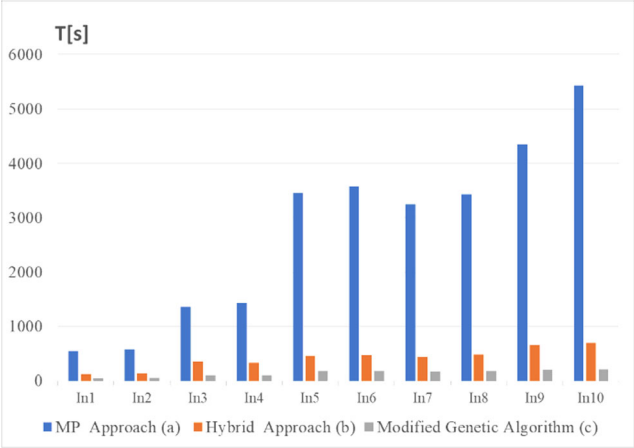

Fig. 16 Computation times of individual methods for a proactive approach (Q5-Table 13)

example). Table 12 presents the received results for the Q_4 question for a single u, whereas Table 13 presents results for the Q_5 question encompassing a sub-set amounting to 80 cases of unavailability of employees u. For all implementations (a), (b), (c), ten experiments were performed for data instances which differed by the number of K tasks, E number of competences, P employees, etc.

It is easy to note that the hybrid approach allows for a reduction of decision variables in the model by approx. 40–60% in relation to the MP approach.

As for the computation time, both methods (b) and (c) turned out to be extremely effective and shorten the time of calculations in case of the Q_4 question (reactive approach) up to 70 times, whereas in case of the Q_5 question (proactive approach) up to 8 times in implementation (b) and up to 25 times in implementation (c) in relation to implementation (a). This is greatly significant in the reactive approach where decisions should practically be made on-line. For the more demanding proactive case, the use of the hybrid approach and dedicated genetic algorithm allows significant reduction in solution time. When analyzing the obtained results in more detail (Tables 12 and 13), it is clear that the application of the MP-based approach (a) is ineffective both in the reactive (Q_4) and the proactive (Q_5) case. It can also be seen that methods (b) and (c) make it possible to obtain a solution in an acceptable time. Especially for larger data instances (Fig. 16), the use of the dedicated genetic algorithm (c) is more promising. This is due to the fact that the computation time does not increase as much with the increase in the size of the data instance as it is in the case of the hybrid approach (b).

5 Conclusions

The proposed model of employee competence configuration may provide a basis to support decisions related to selection and supplementation of the competences of employees engaged in the production process, feasibility tests of a pre-defined production schedule, and to ensure the robustness of the performance of a production schedule to employee absenteeism, holiday planning and employee training, etc. In addition, the model allows you to find the allocation of employees with specific competencies for tasks. Due to the mode of problem formalization and implementation in AMPL (Appendix), it may be solved in an environment of mathematical programming, such as LINGO, SCIP, CPLEX, Gurobi, (https://en.wikipedia.org/wiki/List_of_optimization_software), etc., which confirms its significant universality. Application of the dedicated genetic algorithm to its implementation is very efficient, as was shown during the computational experiments (Table 13, Fig. 16). This efficiency results from the proposed dedicated presolving procedure of the modeled problem (Fig. 8) and the appropriate handling of constraints (repair function, method of coding, etc.).

In the course of further work, implementation of the Q_7 and Q_8 questions is planned, along with the possibility of finding answers to all questions at any moment of T time of schedule performance. Extension of the model is also planned, consisting in introduction of work time limitations of individual employees, as well as a possibility of undertaking performance of tasks partially executed by other employees and fuzzy logic mechanisms [34]. Another area of model extension will consist in introduction of costs of use of employees with specific competences. In future research, it is planned to use the proposed model or its modified versions for project management [22], assembly lines [14], transportation [4], supply chain management [12] and teacher management [29, 30].

Appendix

Implementation model of Employee Competence Configuration Problem in AMPL

```
Description of AMPL files:
z.run - startup file
z.mod -model file
z.dat - data file

Script file z.run
   model z.mod;
   data  z.dat;
   option solver gurobi;
   solve;
   display X;

Model file z.mod
   set Resources;
   set Tasks;
   set Excluded_resources;
   param G{Resources,Tasks};
   param Go{Resources,Tasks};
   param Gk{Resources,Tasks};
   param F{Excluded_resources,Resources};
   param R{Tasks,Tasks}
   param L;
   param St;
   var   X{Excluded_resources,Resources,Tasks} >=0, binary;
   var   Gx{Resources,Tasks} >=0, binary;
   var   Wx{Excluded_resources} >=0, binary;
   var   Y{Excluded_resources,Tasks} >=0, binary;
   minimize cost_chang:
    sum{i in Resources, k in Tasks}
     Gk[i,k]*Gx[i,k];
   subject to C1 {u in Excluded_resources,k in Tasks}:
     sum{i in Resources} X[u,i,k]+Y[u,k] =1 ;
```

```
     subject to C2 {i in Resources, k in Tasks}:
      Gx[i,k]<=Go[i,k]+G[i,k];
     subject to C3 {u in Excluded_resources,i in Resources, k in Tasks}:
      X[u,i,k]<=Gx[i,k];
     subject to C4 {u in Excluded_resources,i in Resources, k in Tasks}:
      X[u,i,k]<=F[u,i];
     subject to C5 {u in Excluded_resources}:
      sum{k in Tasks} Y[u,k]<=St*Wx[u];
     subject to C6:
      sum{u in Excluded_resources} Wx[u]<=L;
     subject to C10 {u in Excluded_resources,
      i in Resources, k1 in Tasks, k2 in Tasks:R[k1,k2]=1}:
      X[u,i,k1]+X[u,i,k2]<=1;
Data file z.dat
data;
set Resources :=i1 i2 i3 i4 i5 i6 i7 i8 i9 i10;
set Tasks     :=k1 k2 k3 k4 k5 k6 k7 k8 k9 k10 k11 k12 k13 k14 k15;
set Excluded_resources :=u1 u2 u3 u4 u5 u6 u7 u8 u9 u10;
param  G:    k1 k2 k3 k4 k5 k6 k7 k8 k9 k10 k11 k12 k13 k14 k15:=
       i1    1  0  0  1  0  1  0  0  0   0   1   0   1   0   0
       i2    1  0  0  1  0  1  0  0  0   0   1   0   1   0   0
       i3    0  1  0  0  0  0  0  1  1   0   0   1   0   0   1
       i4    0  1  0  0  0  0  0  1  0   0   0   0   0   0   0
       i5    0  0  1  0  1  0  1  0  0   1   0   0   0   1   0
       i6    0  0  1  0  0  0  1  0  0   0   1   0   1   0   0
       i7    1  0  0  1  1  1  0  0  0   1   0   0   0   1   0
       i8    0  0  1  0  1  0  1  0  0   1   0   0   0   1   0
       i9    0  1  0  0  0  0  0  1  1   0   0   1   0   0   1
       i10   0  0  0  0  0  0  0  0  1   0   0   1   0   0   1;
param         Go:  k1 k2 k3 k4 k5 k6 k7 k8 k9 k10 k11 k12 k13 k14 k15:=
              i1   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i2   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i3   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i4   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i5   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i6   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i7   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i8   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i9   0  0  0  0  0  0  0  0  0   0   0   0   0   0   0
              i10  0  0  0  0  0  0  0  0  0   0   0   0   0   0   0;
Param         Gk:  k1 k2 k3 k4 k5 k6 k7 k8 k9 k10 k11 k12 k13 k14 k15:=
              i1   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i2   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i3   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i4   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i5   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i6   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i7   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i8   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i9   1  1  1  1  1  1  1  1  1   1   1   1   1   1   1
              i10  1  1  1  1  1  1  1  1  1   1   1   1   1   1   1;
param F:i1 i2 i3 i4 i5 i6 i7 i8 i9 i10:=
     u1  0  1  1  1  1  1  1  1  1   1
     u2  1  0  1  1  1  1  1  1  1   1
     u3  1  1  0  1  1  1  1  1  1   1
     u4  1  1  1  0  1  1  1  1  1   1
     u5  1  1  1  1  0  1  1  1  1   1
     u6  1  1  1  1  1  0  1  1  1   1
     u7  1  1  1  1  1  1  0  1  1   1
     u8  1  1  1  1  1  1  1  0  1   1
     u9  1  1  1  1  1  1  1  1  0   1
    u10  1  1  1  1  1  1  1  1  1   0;
Param R:k1 k2 k3 k4 k5 k6 k7 k8 k9 k10 k11 k12 k13 k14 k15:=
      k1 0  0  0  0  1  0  0  0  1   0   0   0   0   0   0
      k2 0  0  0  0  0  1  0  0  0   1   0   0   0   0   0
      k3 0  0  0  0  0  1  0  0  0   0   0   1   0   0   0
      k4 0  0  0  0  0  0  0  0  0   0   0   0   0   0   1
      k5 1  0  0  0  0  0  0  0  1   0   0   0   0   0   0
```

```
      k6  0  1  1  0  0  0  0  0  0  1  0  0  0  0  0
      k7  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0
      k8  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0
      k9  1  0  0  0  1  0  0  0  0  0  0  0  0  0  0
     k10  0  1  0  0  0  1  0  0  0  0  0  0  0  0  0
     k11  0  0  0  0  0  0  0  1  0  0  0  0  0  1  0
     k12  0  0  1  0  0  0  1  0  0  0  0  0  0  0  0
     k13  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0
     k14  0  0  0  0  0  0  0  1  0  0  1  0  0  0  0
     k15  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0;
param L:=0;
param St:=1000;
```


References

1. Abedinnia H, Glock C, Schneider M, Grosse E (2017) Machine scheduling problems in production: a tertiary study. Comput Ind Eng 111:403–416
2. Ahmadi-Javid A, Hooshangi-Tabrizi P (2017) Integrating employee timetabling with scheduling of machines and transporters in a job-shop environment: a mathematical formulation and an anarchic society optimization algorithm. Comput Oper Res 84:73–91
3. Artigues C, Gendreau M, Rousseau L, Vergnaud A (2009) Solving an integrated employee timetabling and job-shop scheduling problem via hybrid branch-and-bound. Comput Oper Res 36:2330–2340
4. Bocewicz G (2014) Robustness of multimodal transportation networks. Eksploatacja i Niezawodność–Maintenance and Reliability 16(2):259–269
5. Bouajaja S, Dridi N (2017) A survey on human resource allocation problem and its applications. Oper Res Int J 17:339–369. https://doi.org/10.1007/s12351-016-0247-8
6. Brucker P, Qu R, Burke E (2011) Personnel scheduling: models and complexity. Eur J Oper Res 210:467–473
7. Daniels R, Mazzola J (1994) Flow shop scheduling with resource flexibility. Oper Res 42:504–522
8. Daniels R, Mazzola J, Shi D (2004) Flow shop scheduling with partial resource flexibility. Manag Sci 50:658–669
9. Dasović B, Galić M, Klanšek U (2020) A survey on integration of optimization and project management tools for sustainable construction scheduling. Sustainability 12:3405. https://doi.org/10.3390/su12083405
10. Ernst A, Jiang H, Krishnamoorthy M, Sier D (2004) Staff scheduling and rostering: a review of applications, methods and models. Eur J Oper Res 153:3–27
11. Framinan JM, Leisten R, García RR (2014) Manufacturing scheduling systems. An Integrated View on Models, Methods and Tools. Springer-Verlag London, https://doi.org/10.1007/978-1-4471-6272-8
12. Grzybowska K (2019) Reconfiguration to renovation in a sustainable supply chain. Burduk A., Chlebus E., Nowakowski T., Tubis A., Intelligent Systems in Production Engineering and Maintenance, Part of the Advances in Intelligent Systems and Computing book series (AISC, volume 835), Springer Nature Switzerland AG, 761–770
13. Hassannayebi E, Zegordi SH, Yaghini M, Amin-Naseri MR (2017) Timetable optimization models and methods for minimizing passenger waiting time at public transit terminals. Transp Plan Technol 40(3):278–304. https://doi.org/10.1080/03081060.2017.1283156
14. Janardhanan MN, Li Z, Bocewicz G, Banaszak Z, Nielsen P (2019) Metaheuristic algorithms for balancing robotic assembly lines with sequence-dependent robot setup times. Appl Math Model 65:256–270. https://doi.org/10.1016/j.apm.2018.08.016
15. Khojasteh Y (2016) Production control systems. A Guide to Enhance Performance of Pull Systems, Springer Japan. https://doi.org/10.1007/978-4-431-55197-3
16. Kopanos GM, Puigjane L (2019) Solving large-scale production scheduling and planning in the process industries. Springer Nature Switzerland AG. https://doi.org/10.1007/978-3-030-01183-3
17. Lee CKH (2018) A review of applications of genetic algorithms in operations management. Eng Appl Artif Intell 76:1–12. https://doi.org/10.1016/j.engappai.2018.08.011
18. Panik MJ (2018) Linear programming and resource allocation modeling. Wiley
19. Panos F, Paraskevi T, Vassilis G (2018) Industry 4.0: required personnel competences. International scientific conference industry 4.0, at Varna, Bulgaria
20. Pinedo ML (2009) Planning and scheduling in manufacturing and services. Springer-Verlag New York, https://doi.org/10.1007/978-1-4419-0910-7
21. Prifti L, Knigge M, Kienegger H, Krcmar H (2017) A competency model for "Industrie 4.0" employees. Wirtschaftsinformatik (WI) 2017, at St. Gallen, Switzerland
22. Relich M (2017) Identifying project alternatives with the use of constraint programming. Advances in Intelligent Systems and Computing 521:3–13
23. Rossit DA, Tohmé F, Frutos M (2019) Industry 4.0: Smart scheduling. Int J Prod Res 57(12):3802–3813. https://doi.org/10.1080/00207543.2018.1504248
24. Schulze M, Zimmermann J (2018) Staff and machine shift scheduling in a German potash mine. J Sched, 1–22. IFAC ADCHEM Shenyang, Liaoning, China, 2017, July 25-27, 2018 161

25. Sitek P, Wikarek J (2016) A hybrid programming framework for modeling and solving constraint satisfaction and optimization problems. Sci Program, vol. 2016, Article ID 5102616, 13 pages. https://doi.org/10.1155/2016/5102616
26. Sitek P, Wikarek J (2018) A multi-level approach to ubiquitous modeling and solving constraints in combinatorial optimization problems in production and distribution. Appl Intell 48:1344. https://doi.org/10.1007/s10489-017-1107-9
27. Sitek P, Wikarek J (2019) Capacitated vehicle routing problem with pick-up and alternative delivery (CVRPPAD): model and implementation using hybrid approach. Ann Oper Res 273:257–277. https://doi.org/10.1007/s10479-017-2722-x
28. Sitek P, Wikarek J, Nielsen P (2017) A constraint-driven approach to food supply chain management. Ind Manag Data Syst 117:2115–2138. https://doi.org/10.1108/IMDS-10-2016-0465
29. Szwarc E., Bach-Dąbrowska I., Bocewicz G. (2018) Competence management in teacher assignment planning. In: Damaševičius R., Vasiljevienė G. (eds) Information and Software Technologies. ICIST 2018. Communications in Computer and Information Science, vol 920. Springer, Cham https://doi.org/10.1007/978-3-319-99972-2_37
30. Szwarc E, Wikarek J, Gola A, Bocewicz G, Banaszak Z (2020) Interactive planning of competency-Driven University teaching staff allocation. Appl Sci 10:4894. https://doi.org/10.3390/app10144894
31. Touat M, Bouzidi-Hassini S, Benbouzid-Sitayeb F, Benhamou B (2017) A hybridization of genetic algorithms and fuzzy logic for the single-machine scheduling with flexible maintenance problem under human resource constraints. Appl Soft Comput 59:556–573, ISSN 1568-4946. https://doi.org/10.1016/j.asoc.2017.05.058
32. Yassine AA, Mostafa O, Browning TR (2017) Scheduling multiple, resource-constrained, iterative, product development projects with genetic algorithms. Comput Ind Eng 107:39–56. https://doi.org/10.1016/j.cie.2017.03.001
33. Ernst AT, Jiang H, Krishnamoorthy M, Sier D (2004) Staff scheduling and rostering: A review of applications, methods and models, European J Oper Res 153(1):3–27
34. Arkadiusz Gola, Grzegorz Kłosowski (2019) Development of computer-controlled material handling model by means of fuzzy logic and genetic algorithms. Neurocomputing 338:381–392

Paweł Sitek is Associate Professor at the Department of Electrical Engineering, Automatic Control and Computer Science of Kielce University of Technology in Poland. He obtained M.Sc degree in Electrical Engineering from Kielce University of Technology, Poland, Ph.D. from Silesian University of Technology, Poland and D.Sc. degree in Computer Sciences from the Wroclaw University of Technology, Poland. His research includes operation research, constraints programming techniques, production planning and scheduling, discrete optimization, decision support systems and artificial intelligence. He is an author and co-author over 170 manuscripts including: international journals (JCR), chapters in books and conference proceedings. He is an editor and reviewer for a number of international journals and conferences.

Jarosław Wlkarek is Assistant Professor at the Department of Electrical Engineering, Automatic Control and Computer Science of Kielce University of Technology in Poland. He obtained M.Sc degree in Electrical Engineering from Kielce University of Technology, Poland, Ph.D. from Silesian University of Technology, Poland. His research includes operation research, constraints programming techniques, production planning and scheduling, discrete optimization, and decision support systems. He is an author and co-author over 150 manuscripts including international journals (including on JCR), chapters in books and conference proceedings. He is a reviewer for a number of international journals and conferences.