Competence-based assignment of tasks to workers in factories with demand-driven manufacturing

Bożena Staruch¹ · Bogdan Staruch¹

Accepted: 2 February 2021 / Published online: 18 February 2021
© The Author(s) 2021

Abstract

The paper is motivated by real problems concerning tasks assignment to workers in medium-sized upholstered furniture plants managed using the Demand-Driven Manufacturing. Although the methodology was developed for furniture plants it can be applied to other types of production plants. We involve competence coefficients, which describe the level of the worker's skills or capabilities to perform a specific task. The competence coefficients are also used to block the possibility of assigning the given task to a worker that has no skills to do it. Additionally, we involve a dummy worker to the model which guarantees the existence of a solution to the problem. We present and discuss Integer Linear Programming Models for the posted problem that are closely related to the Generalized Assignment Problem. We also discuss the potential use of the presented methodology to solve real-life problems related to production management.

Keywords Tasks assignment · Competence coefficient · Production planning · Demand-driven manufacturing · Lean production · Kanban · Integer linear programming · Generalized assignment problem · Multi-knapsack problem

Mathematics Subject Classification 90B05 · 90B30 · 90-04 · 90C90

1 Introduction

We present the problem of assigning tasks to workers in medium-sized upholstered furniture factories managed using Demand-Driven Manufacturing. This problem is motivated by real-life challenges described by managers in upholstered furniture plants in Poland. The main idea in this paper was formulated in 2012 in cooperation

✉ Bożena Staruch
bostar@matman.uwm.edu.pl

Bogdan Staruch
bstar@uwm.edu.pl

¹ University of Warmia and Mazury, Olsztyn, Poland

with ITM¹, which provides production management systems in furniture factories. IT implementation of the solution to the presented problem was awarded in 2016 with the ‘Debut of the Year’ award given by the *Rzeczpospolita* newspaper. The system based on this solution works, for example, in Primavera Furniture² factory.

Demand-Driven Manufacturing (details in Sect. 6), typically used in the factories under consideration, requires a high degree of flexibility in production planning. Some aspects such as unpredictable production disturbances, poor health of the working person, or hidden defects in materials, which can only be noticed while performing the task can be only seen during the execution of the task, require immediate reaction and reallocation. The problem of reallocation was considered, for example, in Jain and Elmaraghy (1997) and Strickera and Lanza 2014.

There is a great need to create an effective, flexible and reactive system supporting production management, in particular, human resources management. As far as upholstered furniture factories are concerned, the two stages of production: covers sewing and upholstering most need such a solution.

Besides, the analyzed upholstered furniture factories usually apply the color-coded task priorities (red, yellow, green) based on the Kanban philosophy (see Hammarberg and Joakim 2014). We do not consider any preferences in the stages of production. The proposed methodology applies to every stage independently however, such kinds of preferences are included in the colors of the tasks. Any green task on the given stage of production is assumed to be already produced on the antecedent stages. Every red task has to be finished to use it in the next stages.

We propose different integer linear optimization models for solving the problem of competence-based tasks assignment to workers. Competence coefficients further enhance the model with their ‘regular’ values from an interval $[\min C, \max C]$ that describe the skills or capabilities of each employee to perform each specific task. A subjective appraisal may determine the competence coefficients by the superiors or by some automatic evaluation systems. Alternatively, we can use a mixture of both. It is worth considering the development of an automatic classification model (more on classifiers can be found in Artiemjew 2008; Staruch 2017; Staruch and Staruch 2017). Regardless of the method of evaluation, the values of these factors are to be such that a lower value of the coefficient indicates the greater predisposition of the worker to perform a given task. Thus the competence coefficients play the role of the cost function used in many optimization problems. By setting the values of some competence factors on some high level, we obtain a way of blocking the allocation of tasks to workers with insufficient competence to perform them. On the other hand, the allocation of simple tasks to highly qualified workers can be blocked the same way. The need for blockages is discussed in the domain called *task assignment*, where binary values of competence coefficients indicating skills or authorization are used (see, e.g., IBM Knowledge Center 2019).

An additional and crucial element of the presented model is a dummy worker (named *Dummy* for short) with its competence value treated as a penalty for unfinished tasks. Involving Dummy ensures the existence of a solution for the modeled problem.

¹ ITM Sp. z o.o. Software Company in Olsztyn.

² <https://itm.com.pl/en/primavera-2/>.

Further analysis of tasks allocated to Dummy allows deciding if reallocation of tasks should be performed.

The paper is organized as follows. We describe the elements of the problem in Sect. 2. Section 3 is devoted to the description of the role and the form of the competence coefficients. We present optimization ILP models for solving the main problem of this paper in Sect. 4. We discuss their advantages and disadvantages and compare them with the known problems such as of Generalized Assignment Problem and Multi-Knapsack Problem. The proposition of an algorithmic solution is presented in Sect. 5. Section 6 is devoted to discussion of the most important aspects concerning Demand-Driven Manufacturing. We indicate the most important factors influencing the quality of production planning in the form of parameters describing our solution, including modeling of color-coded task priorities. We show how modification of these parameters allows us to solve problems related to production planning effectively and to react to unexpected events.

2 Conventions and assumptions

We use the abbreviation CBA to indicate the title problem called Competence-Based Assignment of Tasks to Workers.

Assume that there are workers in a plant that are planned to perform a set of tasks, e.g., sewing covers or upholstering furniture. Let us use the following notations:

1. m_0 is the number of *workers*, who are indicated by indices $i = 1, \dots, m_0$;
2. every worker i has his/her *available time* denoted by L_i ;
3. n is the number of tasks planned to be performed, which are indicated by indices $j = 1, \dots, n$;
4. every item j has its fixed normative execution time t_j , named *normative time*;
5. each worker may perform any number of tasks within her/his available time;
6. every task can be performed by exactly one worker;
7. each worker i has specific competencies to perform the task j . The level of these competencies is 'normally' presented in the form of a competence coefficient $C_{ij} \in [\min C, \max C]$ so that the higher the competencies, the lower the competence coefficient value.

The solution to the problem of assigning tasks to workers based on competencies is to assign tasks to workers with a fixed available time so that the total cost of completing the tasks is the 'lowest.'

Notice that the CBA problem does not cover interrelations between different stages of production. Changeover times are not considered too, however, the proper use of competence factors (see Sect. 3) gives some 'seriality' effects.

3 Competence coefficients

Competence coefficients play a fundamental role in the CBA problem. The right choice of these coefficients has a direct impact on higher production efficiency. For

this reason, a well-thought-out system for determining competence factors, tailored to the needs of the production department in the factory, is needed. The role of the competence coefficient is to quantify the skill level of the worker, resulting in faster work and higher quality of task performance.

Competence coefficients can be interpreted as a cost (the lower value, the better), so we can think of the competence coefficients as a cost function, which for the task-worker pair i, j takes the cost value C_{ij} .

Interviews with production managers in upholstered furniture factories show that it is standard practice to assign tasks by hand. The shift manager decides on the allocation of tasks based on his own experience and knowledge of the skills of his subordinates.

There is no one-size-fits-all principle to determine which values of competence coefficients are the best. One factory opts for the quality of performance even at the expense of time, while for the other, time reduction is the most important. In the latter case, the competence factors can express a percentage saving of time for each task and hence we can assume that a reasonable range of competence coefficients is between $\min C = 0.9$ and $\max C = 1$.

Competence coefficients can be used to obtain an assignment of tasks that meet 'seriality' requirements. We understand seriality as allocating a series of the same tasks to a worker, if possible. This seriality saves time by reducing the time it takes to adjust the workstation (changeover time), as well as automation and personal optimization as a consequence of repeating the same tasks many times over. Although the proposed solution does not include any changeover time, tests show that more significant variation in the values of competence coefficients increases the 'seriality' effect. We can obtain the variation mentioned above by taking, for every single task, a large number of different values of competence coefficients, e.g., by specifying these values to three or four decimal places.

The attempt to automate the determination of competence coefficients should start with recording the knowledge of production managers in numerical form. Then the factors affecting the quality of work should be identified, and their numerical evaluation should be recorded in the form of an information table during long-term observation. For example, we quickly identify three of these factors: the number of tasks of a given type carried out so far, the reduction in lead time, the number and importance of corrections (loss of time for corrections, material losses). With such an information table, an appropriate classification model can be used to determine competence coefficients. The obtained coefficients should be monitored on an ongoing basis and, if necessary, corrected.

Machine-learning-based classification methods can be used to solve the problem of determining competence coefficients. There are many models of classifiers in the literature, for example (Artiemjew 2008; Staruch 2017; Staruch and Staruch 2017).

3.1 Dummy worker

The next important element in solving the CBA problem is a fictitious worker, whom we call *Dummy*. Dummy is the last worker (indicated by d) with unlimited available time. Dummy guarantees the existence of a solution and it is used to identify tasks

that have not been assigned. Minimization of a total normative time of tasks assigned to Dummy is the desired property in the solution of CBA. Dummy's competence coefficient C_d is the same for all the tasks and plays the role of penalty for unassigned tasks.

3.2 Task blockade

By setting the values of some competence coefficients at a sufficiently high level (indicated by B), we obtain a way of blocking the allocation of tasks to workers with insufficient competences to perform them. On the other hand, it is also possible to block the allocation of simple tasks to highly qualified workers. The need for blockages is discussed and analyzed in a field called Task Assignment, where binary values of competence coefficients indicating skills or authorization are used (see, e.g. [26]).

The value of B should be greater than C_d . $B = 3 \cdot C_d$ is sufficient to block the assignments as it is the known routine in the classic Transportation Problem.

4 Optimization models

In this section, we discuss various Integer Linear Programming (abbr. ILP) models that can be used to solve the CBA problem.

We use binary decision variables x_{ij} , where $x_{ij} = 1$ if and only if the j th task is assigned to the i th worker. We have $m = m_0 + 1$ constraints describing the requirement that each worker can perform any number of tasks within his or her available time L_i , where Dummy is the m th worker with $L_m = \sum_{j=1}^n l_j$. We also have n constraints saying that each task can be performed by exactly one worker and $m \cdot n$ binary constraints on x_{ij} .

We consider the following two different optimization models $M1$, $M2$.

$$\begin{array}{ll}
 \text{Minimize} & F = \sum_{i=1}^m \sum_{j=1}^n t_j \cdot C_{ij} \cdot x_{ij} \\
 M1: & \\
 \text{Subject to} & \sum_{j=1}^n x_{ij} \cdot t_j \leq L_i, \sum_{i=1}^m x_{ij} = 1, x_{ij} \in \{0, 1\} \\
 \text{Minimize} & F = \sum_{i=1}^m \sum_{j=1}^n t_j \cdot C_{ij} \cdot x_{ij} \\
 M2: & \\
 \text{Subject to} & \sum_{j=1}^n x_{ij} \cdot t_j \cdot C_{ij} \leq L_i, \sum_{i=1}^m x_{ij} = 1, x_{ij} \in \{0, 1\}
 \end{array}$$

The above models differ in the constraints on workers' available time. In model $M1$, we assume that competence coefficients do not affect the speed of workers' work and therefore we plan to complete each task in its normative time. $M2$ presents the assumption that the execution time of the tasks decreases proportionally to the values of competence coefficients. Which model to choose to be implemented in production must be decided by the factory management.

At first glance, we can see that the proposed models are very similar to the minimization problem called in the literature the Generalized Assignment Problem (abbr. GAP) that is known to be NP-hard and even APX-hard to approximate it. There are many

algorithmic methods in the literature to obtain a sub-optimal solution to this problem based on different types of heuristics (see Baykasoglu et al. 2007; Cattrysse et al. 1994; Chu and Beasley 1997; Diaz and Fernandez 2001; Lourenço and Serra 2002; Nauss 2003; Osman 1995; Randall 2004; Savelsbergh 1997; Yagiura et al. 2006).

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^m \sum_{j=1}^n \text{Cost}_{ij} \cdot x_{ij} \\ \text{GAP:} \quad & \\ \text{subject to} \quad & \sum_{j=1}^n x_{ij} \cdot W_{ij} \leq L_i, \sum_{i=1}^m x_{ij} = 1, x_{ij} \in \{0, 1\} \end{aligned}$$

The GAP has the two 2-dimensional functions: the cost function Cost and the weight function W .

$M1$ is equivalent to the GAP when we take $\text{Cost}_{ij} = t_j \cdot C_{ij}$ and $W_{ij} = t_j$. If $\text{Cost}_{ij} = t_j \cdot C_{ij}$ and $W_{ij} = t_j \cdot C_{ij}$ we get that $M2$ is a special instance of the GAP, where $\text{Cost}_{ij} = W_{ij}$.

4.1 The use of dummy in optimization models

Another approach to optimization of the CBA problem is the minimization of the total assignment to Dummy. Practical applications justify the use of this type of model because the tasks that are assigned to Dummy should be treated as tasks that will not be performed during the current production shift. Therefore, the model mentioned above pursues a simple objective formulated as 'to assign as many tasks as possible to the workers.' As Example 1 shows, the value of C_d should be big enough to obtain both: assigning tasks to workers with the best competencies and minimizing Dummy's assignment.

Hence the objective takes the form of minimization of the sum $D = \sum_{j=1}^n t_j \cdot x_{mj}$. This objective function does not take into account the competence factors and the assumed blockages for assigning some tasks to some workers, as well. Hence for each blocked assignment (with indices ij) we set $x_{ij} = 0$ and add these equations to the set of constraints. Let $DM1$ and $DM2$ denote the described above Dummy optimization models with constraints corresponding to $M1$ and $M2$, respectively.

It is easy to see that an equivalent formulation of $DM1$ and $DM2$ is replacing the objective function from these models by maximization of the $D^* = \sum_{i=1}^{m_o} \sum_{j=1}^n t_j \cdot x_{ij}$. The symbols D^*M1 and D^*M2 denote these two models. Notice that Dummy is not represented in these two models, as we have no effective constraints on the Dummy's available time as well as Dummy is not included in the objective. Thus the optimization objective is to choose the tasks from the set of all tasks that give maximal profit (total normative time of assigned tasks). Hence, D^*M1 and D^*M2 are instances of the Multi-Knapsack Problem (abbr. MKP) (see Martello and Toth 1990), where the profits are set as $p_{ij} = t_j$ and the weights are described as $W_{ij} = t_j$ for $M1$ and $W_{ij} = t_j \cdot C_{ij}$ for $M2$. Thus we can use some of the sub-optimal algorithms for solving MKP from the known methods described in the rich literature on the subject.

To get the reasonable models $DM1$, $DM2$, D^*M1 , and D^*M2 , we need the assumption that there are enough tasks so that the optimal solution for $M1$, $M2$ includes the nonempty set of tasks assigned to Dummy.

Notice that $DM1$ (and D^*M1 as well) does not include any competence factors, therefore it is useless for solving CBA with differentiated values of competence factors.

Now, we discuss the issue of determining the value of the Dummy's competence coefficient C_d by comparing the $DM2$ model with the $M2$ model. First, it should be noted that they are not equivalent.

Obviously, in the case when all the tasks can be assigned to the 'regular' workers (i.e. $D = 0$) it is enough to take $C_d > \max C$ to be sure that any optimal solution for $M2$ will not assign any task to Dummy. On the other hand, in this situation any solution for $DM2$ with $D = 0$ is optimal without guaranteeing optimal solutions for $M2$.

Example 1 Consider the CBA problem with parameters as in the following table:

Then we get the optimal solution for $DM2$ assigning $Task_1$ and $Task_2$ to Dummy, $Task_3$ to the first worker and $Task_4$ to the second worker. Then $D_0 = 90$, $F_0 = 117 + 100 + 90 = 307$.

The optimal solution for $M2$ is the assignment of $Task_3$ to Dummy, $Task_4$ to the first worker and the rest of the tasks to the second worker. In this case $D = 100$, $F = 130 + 90 + 40 + 45 = 305$.

Hence, $D > D_0$ while $F < F_0$. Notice, that a greater value of C_d can eliminate this undesirable property. In this example, $C_d \geq 1,5$ assures that $F \geq F_0$.

Tasks	Workers			
	t_j	C_{1j}	C_{2j}	C_d
$Task_1$	40	0.9	1	1.3
$Task_2$	50	1	0.9	1.3
$Task_3$	100	1	1	1.3
$Task_4$	100	0.9	0.9	1.3
L_i :		100	100	290

This example shows that $M2$ and $DM2$ are not equivalent and that the value of C_d has an impact on solutions and relationships between different kinds of models.

Experiments carried out in plants mentioned in this paper showed that it is enough to take $C_d = 2\max C$.

5 An exemplary algorithmic solution

The proposed algorithmic solution is useful for finding an approximate solution to both problems $M2$ and $DM2$. It means that the algorithm tries to assign tasks to workers with the best competencies while minimizing the Dummy's assignment.

The first step of the algorithm named INITIAL is to obtain an initial solution. The greedy algorithm takes the tasks according to the decreasing order of normative times and assigns them to the workers with the lowest possible competence coefficient.

We assume that the tasks are sorted by decreasing normative time. We associate to every worker i his/her remaining time R_i , where $R_i = L_i$ at start. Then for every

$i = 1, \dots, m$ we search the i th row of the competency matrix and for the minimal value, say c_{ij} and check if $c_{ij} \cdot t_i \leq R_i$. If the answer is 'Yes,' we associate the j th task to the i th worker and update her/his remaining time $R_i := R_i - c_{ij} \cdot t_j$ and then go to the next task. If the answer is 'No,' we search for the next minimal value in the i th row and repeat this sequence of activities until all the tasks are assigned to workers, Dummy included. As a result, we get the assignment of tasks to workers, which we call the *initial solution*.

The second step called IMPROVE is to improve the initial solution. We suggest using a procedure that minimizes the total assignment to the Dummy while optimizing the objective function for $M2$. However, a slight modification (i.e. taking $t_i \leq R_i$ instead of $c_{ij} \cdot t_i \leq R_i$) results in solving for $M2$.

Let's assume that the tasks assigned to the Dummy are sorted by decreasing normative times. The two main functions TWO-FOR-ONE and ONE-FOR-ONE are used at this stage of the algorithm. These functions have two tasks or one task currently unassigned and try to assign them the chosen worker in the place of other task. Such a reassignment should improve the solution. In case of TWO-FOR-ONE function let us assume that we have the two tasks j, k with $t_j \geq t_k$ which are taken from Dummy. If we assign them to the worker i in place of the task l , then the value of the objective function is changed and the difference will be called the improveValue and $\text{improveValue} = C_d \cdot (t_j + t_k) - (c_{ij} \cdot t_j + c_{ik} \cdot t_k) + c_{il} \cdot t_l - C_d \cdot t_l$. We try to maximize the improveValue under assumptions that that

Cond1. $t_l < t_j + t_k$ *Cond2.* $c_{ij} \cdot t_j + c_{ik} \cdot t_k \leq R_i + c_{il} \cdot t_l$.

The ONE-FOR-ONE function takes a one task t_j with $\text{improveValue} = C_d \cdot (t_j) - c_{ij} \cdot t_j + c_{il} \cdot t_l - C_d \cdot t_l$

Cond1. $t_l < t_j$

Cond2. $c_{ij} \cdot t_j + c_{ik} \leq R_i + c_{il} \cdot t_l$.

Thus the IMPROVE stage is as follows:

1. Take the first two tasks from Dummy j, k and search for a worker i with the task l so that the conditions *Cond1.* and *Cond2.* are satisfied and the improveValue is maximal.
2. If such a worker exists, we assign the tasks j, k to him/her, take off the l task and update the remaining time $R_i = R_i - c_{ij} \cdot t_j - c_{ik} \cdot t_k + c_{il} \cdot t_l$. Now, we loop the ONE-FOR-ONE function with the current task l , possibly substituting it by other current tasks. If no substitution is done for the current task the loop stops and the task goes to Dummy and takes place according to its normative time.
3. If after searching the entire set of workers (using the TWO-FOR-ONE function) no substitution is done, the task j goes to Dummy to the first place, and the task i goes to the new list of unassigned tasks.
4. While there are at least two tasks assigned to Dummy we repeat steps 1. - 3. Finally, Dummy has one task only. This task goes to the new list of rejected parts.
5. Now, all the unassigned tasks are assigned to Dummy, and we can repeat the whole procedure until the list of unassigned tasks is empty. Optionally, we can stop the algorithm and assign all the tasks from unassigned tasks and from rejected tasks to Dummy.

6. An additional option is to perform the ONE-FOR-ONE function for every rejected task, at the end of the algorithm.

5.1 Experimental results

This algorithm is not optimal, but it is fast, so in the absence of a better solution it can be successfully used. The computational complexity of the INITIAL stage is $n \log n$ for sorting and $n \cdot m$ for assigning. Let n_0 denote the number of tasks assigned to Dummy. The IMPROVE stage takes pairs of tasks assigned to Dummy, which is approximated by $O(n_0^2)$. Search procedures in steps 1.- 3. are approximated by $O(n^2)$.

The proposed solution is also useful in managing color based priority. First, we run the INITIAL stage for all the tasks and then perform the IMPROVE stage for the 'red' tasks assigned to Dummy performing steps 1. - 5. If there is any 'red' task in rejected tasks we perform ONE-FOR-ONE function for 'red' rejected tasks waiving *Cond.1* and taking off 'non-red' tasks, only.

We carried out a series of experiments for data sets that were described by some chosen parameters. All the tests were made on personal notebook (Intel(R) Core(TM) I5-2520 M CPU @ 2.50 GHz 2501 MHz) The number of workers was 10, 30, and 50. The available time for every worker was fixed to 450 minutes. The total normative time of tasks was expressed as the total workers' time multiplied by the given factor ($F1 = 1.25$ or $F2 = 1.5$) and next, the set of tasks was generated. We used the following two sequences $S1 = (0.2, 0.6, 0.2)$, $S2 = (0, 0.2, 0.8)$ to describe the percentage share of long, medium and short tasks, where normative times for long tasks were drawn randomly from 60 to 120 minutes, for medium tasks from 30 to 60 minutes, and short tasks from 10 to 30 minutes. The competence factors were randomly drawn from 0.9 to 1 with an accuracy of 0.01. The number of blocked pairs "task-worker," i.e. pairs with the competence factor equal to 3, was on the level of 10 percent of the total time of tasks. The competence factor for Dummy is fixed on 2.

We use notation like $50F1S2$ to describe a data set obtained for 50 workers with the $F1$ factor and the $S2$ sequence.

We compared our algorithm with the algorithms (cut methods and branch methods) dedicated to solving ILP problems and implemented in the GNU Linear Programming Kit (abbr. GLPK) GLPK 2020. The first one is based on Tomlin-Driebeek heuristic Driebeek 1966 (GLPK parameter `-drtom`), and the second one is based on different cut methods (GLPK parameter `-cuts`). For saving time we set the parameter `-tmlim` to 900 seconds. The experimental results are presented as an average percentage gap, where the percentage gap was computed as the deviation of the obtained value of the objective function from the lower bound of the instance divided relative to the lower bound. The lower bound for the given instance is the value of the objective function for continuous LP relaxation.

The table below contains the experimental results, where the first two columns describe the used data sets and the number of instances. The next three columns contain results for the algorithm presented in this paper: average runtime of the algorithm, average percentage gap for the INITIAL stage, and average efficiency after the IMPROVE stage. The last three columns present average percentage gaps for the two ILP methods

Table 1 Experimental Results - Summary

Data Sets		Algorithm			GLPK		
Descrip.	No.	Runtime	INIT	IMPR	–drtom	–Cuts	Fails
10F1S2	10	< 10s	2.9	1.9	0.8	0.8	0
10F2S2	10	< 10s	3.5	2.7	0.8	0.9	0
30F1S1	10	< 50s	2.5	1.5	2.1	1.5	40
30F2S1	10	< 100s	4.1	2.3	4.7	3.6	50
50F2S1	10	< 1000s	2.5	1.7	1.7	1.6	50

and also, a share of fails, where the failure means that after 1000 seconds the given algorithm got no solution.

The GLPK methods gave non-optimal results in all the cases and for instances with a rather big number of tasks they often fail. These instances are hard also for our algorithm, where the runtime of the IMPROVE stage is from 900 to 1000 seconds. On the other hand, the INITIAL stage took less than one second giving quite good results.

6 Application of CBA in production planning

This section is devoted to discussion of application of the CBA problem in real-life Demand Driven Institute 2019. Demand-Driven Manufacturing is a modern method of demand and production management that is associated to a variety of managements methods like Lean (or Just-in-Time) Manufacturing (Ohno 1988; Womack and Jones 2003), Kanban (Anderson 2010; Hammarberg and Joakim 2014; Ladas 2009, Scotland 2018), Product Customization (Hvam et al. 2008), Product Differentiation (Sharp and Dawes 2001).

First, we describe the main characteristics of production management under consideration.

1. *Product variety* - applied routines like Product Customization or Product Differentiation strengthen this effect
2. *Short customer tolerance times* - a short time to complete the order, deliver the product on time and meet deadlines at all stages of production
3. *On-demand production* - all products are manufactured after acceptance of demand, no production for stocks
4. *Pressure to less inventory* - production planning to save storage space, both for raw materials and finished products
5. *Priorities indicating which products should be made first* - the most common routine is Kanban based system of the three colors 'Red', 'Yellow,' 'Green.'
6. *The need for flexibility in production planning* - fast reaction on different kinds of changes and perturbation in production

The first aspect called 'product variety' is the main characteristic of manufacturing in plants under consideration. Such a diversity of production affects the size of the

CBA problem. However, we claim that CBA is a useful and flexible tool for solving most problems concerning such a kind of production.

The proposed methodology is based on repeated use of the algorithm solving the CBA problem. Therefore, we assume that we have a fast algorithm (abbr.ALG) for solving CBA. There is a variety of sub-optimal algorithmic solutions for GAP or MKP that can be found in the literature. A decision which solution is the most useful in the application of CBA in production planning should be based mainly on the 'speed' criterion. Because of high 'product variety,' a fast algorithm is more desirable even at the expense of accuracy. In the absence of better algorithms, the solution proposed in the previous section can play role of ALG.

CBA problem is described by parameters: number of tasks, normative times of tasks, number of employees, available times of workers, competence matrix. We consider also additional information related to the priority of tasks in the form of color-coding, for example, as in the Kanban method. Red color means that the tasks are urgent and must be completed within the fixed deadline. Yellow indicates tasks that can only be carried out if the red tasks have been assigned in advance. Green indicates a task that can be performed after red and yellow tasks are assigned. You can also consider a preference model that takes into account a more significant number of degrees of preference than the three ones described above.

We are going to show how to use the CBA (supported by ALG) for solving problems caused by the aspects 2.- 6. above. The general schema is as follows:

- 1. Input Data** We set the values of parameters of CBA basing on the aim of the use of CBA and the use of color-coded priority
- 2. Run ALG** We run ALG for the Input Data
- 3. Analysis of the Output of ALG** We analyze the result of ALG, and we change the Input Data and run ALG again if we need it.

To use CBA to set the date of order completion (aspect 2. 'short customer tolerance times'), we need to set a certain date of completion and flag the previously planned tasks to red, and the new tasks to yellow. If Dummy has no red task in the output of ALG, then the new demand can be considered as being planned on this date. If not, we check the next date of completion.

The aspects 3–5 are reflected in the appropriate prioritization of tasks. Since storing space is minimal, there are few yellow tasks and even fewer green tasks. The typical situation is that production plans are rather tight. Hence the fast reaction on production disturbances (aspect 6.) is necessary. We can run ALG with input data updated after disturbances. If Dummy has any red task assigned to, the manager can react in the following different ways basing on the above general schema:

1. Add a new worker or increase the workers' available times.
2. Change competence factors, even by removing the blockades
3. Assign the red tasks first, fix the solution by subtraction the assigned tasks and reduce the available workers' time, and assign the rest of tasks
4. Finally, some red tasks may be treated as yellow

7 Summary

We discussed the problem of competence-based tasks assignment to workers in factories that use Demand-Driven Manufacturing. The proposed solution to this problem is based on our interviews with managers in medium-sized upholstered furniture factories in Poland. Furniture production is one of the most important industries in Poland, and there is a need to implement some innovative solutions supporting production management. However, there is no reason preventing CBA to be applied to any type of production plants.

A part of the presented CBA problem was awarded in 2016 with the ‘Debut of the Year’ award given by the *Rzeczpospolita* newspaper. We then introduced the coefficients of competence, blockage and virtual Dummy worker. The algorithmic solution based on the continuous relaxation of the *M2* model and the repetition of the algorithm in rounds was far from perfect. However, it resulted in such an improvement in the quality of assignment of tasks to the workers that it met with appreciation in the form of verbal comments and the award mentioned above. The solution called VINCI is implemented, for example, in Primavera Furniture³ factory.

This paper is based on a more in-depth formal analysis of the real-life production challenges. We discussed the application of the CBA model in production planning, where the main advantage of our solution is its flexibility. This flexibility has been achieved by highlighting the most critical elements (parameters) influencing the final assignment of tasks to the workers.

The optimization models described in this paper indicate the optimization areas (Generalized Assignment Problem, Multi-Knapsack Problem) which, as they have been known for years, offer many different algorithms to solve them. Thus we can choose the best algorithmic solution that meets the managers’ requirements, taking into account the specific way of production management, the practices prevailing in the factory, the specificity of production, directions of optimization.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Anderson DJ (2010) Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press. ISBN 0-9845214-0-2
- Artiemjew P (2008) On classification of data by means of rough mereological granules of objects and rules. LNAI 5009. Springer, Berlin, pp 221–228

³ <https://primaverafurniture.com/en/>.

- Baykasoglu A, Özbakir L, Tapkan P (2007) Artificial bee colony algorithm and its application to generalized assignment problem. In: Chan FTS, Tiwari MK (eds) *Swarm intelligence: focus on ant and particle swarm optimization*. I-Tech Education and Publishing, Austria, pp 113–144
- Cattrysse D, Salomon M, Van Wassenhove LN (1994) A set partitioning heuristic for the generalized assignment problem. *Eur J Op Res* 72:167–174
- Chu PC, Beasley JE (1997) A genetic algorithm for the generalized assignment problem. *Computer Op Res* 24:17–23
- Demand Driven Institute, (2019) <https://www.demanddriveninstitute.com>, last accessed 2019/03/25
- Diaz JA, Fernandez E (2001) A tabu search heuristic for the generalized assignment problem. *Eur J Op Res* 132:22–38
- Driebeek Norman J (1966) An algorithm for the solution of mixed integer programming. *Manag Sci* 12:576–587
- Hammarberg M, Joakim S (2014) *Kanban in action*. Manning Publications, Shelter Island, NY ISBN 978-1-617291-05-0
- GLPK (2020) <https://www.gnu.org/software/glpk/>, last accessed 2020/03/07
- Hvam L, Mortensen NH, Riis J (2008) *Product Customization*, Springer, ISBN: 978-35407144842008
- IBM Knowledge Center, (2019) <https://www.ibm.com/support/knowledgecenter/en>, last accessed 2019/03/25
- Jain AK, Elmaraghy HA (1997) Production scheduling/ rescheduling in flexible manufacturing. *Int J Prod Res* 35(1):281–309
- Ladas C (2009) *Scrumban: essays on Kanban systems for lean software development*. Modus Cooperandi Press, California ISBN 978-0578002149
- Loureço HR, Serra D (2002) Adaptive search heuristics for the generalized assignment problem. *Mathw Soft Comput* 9:209–234
- Martello S, Toth P (1990) *Knapsack problems: algorithms and computer implementations*. Wiley, Chichester
- Nauss RM (2003) Solving the generalized assignment problem: an optimizing and heuristic approach. *Inform J Comput* 15:249–266
- Ohno T (1988) *Toyota production system: beyond large-scale production*. CRC Press, Ohio ISBN 978-0-915299-14-0
- Osman IH (1995) Heuristics for the generalized assignment problem: simulated annealing and tabu search approaches. *OR Spektrum* 17:211–225
- Randall M (2004) Heuristics for ant colony optimisation using the generalised assignment problem, In: *Proceedings of IEEE Congress on Evolutionary Computation*, Portland, Oregon, USA, pp. 1916–1923
- Savelsbergh M (1997) A branch-and-price algorithm for the generalized assignment problem. *Op Res* 45:831–841
- Scotland C (2018) <http://www.methodsandtools.com/archive/archive.php?id=104>. last accessed 2018/10/21
- Sharp B, Dawes J (2001) What is differentiation and How Does it work? *J Mark Manag* 17:739–59. <https://doi.org/10.1362/026725701323366809>
- Staruch B (2017) Classification model based on topological approximation space. *LNCS* 10314:570–578
- Staruch B, Staruch B (2017) A topological approximation space based on open sets of topology generated by coverings. *LNCS* 10314:130–137
- Strickera N, Lanza G (2014) The concept of robustness in production systems and its correlation to disturbances. *Procedia CIRP* 19:87–92
- Womack JP, Jones DT (2003) *Lean thinking: banish waste and create wealth in your corporation*. First Free Press, Sweden ISBN 0-7432-4927-5
- Yagiura M, Ibaraki T, Glover F (2006) A path relinking approach with ejection chains for the generalized assignment problem. *Eur J Oper Res* 169:548–569