**Journal homepage:** https://itce.com.ua/en

**Information Technologies and Computer Engineering** **Vol. 22, No. 2. 2025**

**Article's History:** Received: 04.03.2025 Revised: 13.06.2025 Accepted: 28.08.2025

UDC 519.8 + 004.8 DOI: 10.31649/vitce/2.2025.85

# Scalable mathematical model for distribution of production orders

**Kostiantyn Hrishchenko***

Postgraduate Student
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
03056, 37 Beresteyskyi Ave., Kyiv, Ukraine
https://orcid.org/0009-0008-9251-0222

**Oleksii Pysarchuk**

Doctor of Technical Sciences, Professor
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
03056, 37 Beresteysky Ave., Kyiv, Ukraine
https://orcid.org/0000-0001-5271-0248

**Abstract.** The article proposed a mathematical model of the optimisation problem of planning production orders under conditions of limited resources. The growth in production variability necessitates highly productive and efficient algorithms and models that are capable of processing large numbers of orders and are highly adaptable to new criteria, rules and factors. The aim of the study was to synthesise a scalable mathematical model that takes into account the rules for constructing schedules and allows solving large-scale problems within acceptable time limits. The study proposed a discrete mathematical model that includes a system of constraints and an optimisation criterion. The model meets the real requirements of production, in particular, it takes into account the order of processing tasks within a single order and makes it impossible to perform tasks simultaneously on one resource. To find solutions to the model, the CP-SAT algorithm was used, which is part of the Google Or-Tools package and is recognised as one of the most productive algorithms for discrete optimisation tasks. For input data sets ranging from 5 to 40 dimensions, measurements were taken of the time and space costs of solving the created model and finding the optimal solution. In order to improve the scalability of the approach, an iterative model for distributing orders in controlled-size parts was developed. In it, the size of the subtask is determined by a parameter that limits the number of orders included in the base model, allowing the computational complexity to be adjusted. The results of the experiments showed that the scalable model provided an effective solution to problems with a dimension of 60 orders in a time acceptable for this type of system. A comparison of the basic model and the improved model showed a reduction in time consumption and memory consumption for large-scale tasks. At the same time, linear scaling was observed in both calculation time and memory consumption as the number of orders increased, which guarantees the effective application of this model for larger scales as well. The results of the research laid the foundation for the practical application of the developed scalable model in production planning information systems at enterprises with high loads and large order volumes

**Keywords:** optimisation; constraint programming; discrete linear programming; mathematical model; resource allocation; planning

## Introduction

Resource allocation and order fulfilment planning are the most important and, at the same time, the most difficult tasks required of modern production management systems. The competitive advantage of an enterprise largely depends on the effectiveness of planning. Under current circumstances, improving productivity and reliability in solving large-scale resource allocation and planning tasks will expand the possibilities of applying information systems, enabling the automation of resource allocation and planning for enterprises and organisations that were previously limited due to the inefficiency of existing technologies. Resource allocation and planning belong to the

**Suggested Citation:**
Hrishchenko, K., & Pysarchuk, O. (2025). Scalable mathematical model for distribution of production orders. *Information Technologies and Computer Engineering*, 22(2), 85-95. doi: 10.31649/vitce/2.2025.85

*Corresponding author

problems of combinatorial optimisation. Optimisation aims to find the extrema of the objective function on a set of feasible solutions defined by constraints. A key element of optimisation tasks is a mathematical model, which must take into account the existing rules for forming a schedule, commonly referred to as constraints. It is important that the requirements for the range of technologies that can be applied to solve the problem are laid down at the stage of model development and formalisation.

Constraint programming, along with integer linear programming, is one of the classic methods for solving planning and resource allocation problems. In the work of B. Ofoghi *et al*. (2020) proposed approaches to the development of integer models for applied problems. The main types of constraints and objective functions were described and structured. The developed structure made it possible to systematise the approach to modelling, as demonstrated in the problems of planning, supply chain optimisation, and routing. In contrast to formal models, J.L. Andrade-Pineda *et al*. (2020) proposed a new heuristic algorithm for a highly specialised planning problem. The advantage of heuristic calculations in terms of productivity while maintaining the high quality of the calculated plans was demonstrated. In combination with heuristics, K. Musiał *et al*. (2023) applied the annealing simulation method. This combination improved the quality of the calculated plans, emphasising the importance of using hybrid methods.

There is also growing interest in the application of machine learning methods to optimisation problems. M. Zhang *et al*. (2022) applied deep reinforcement learning to the dynamic planning problem. The trained PPO (Proximal Policy Optimisation) model with a selected set of hyperparameters demonstrated high efficiency, surpassing other approximate approaches in terms of performance and planning quality. Despite this, constraint programming remains relevant due to its ability to find global optima and its formal, deterministic nature. Based on these properties, as well as the flexibility of the approach, F. Çetinkaya *et al*. (2021) applied an integer linear model for rapid hypothesis testing. For small and medium-sized problems, the model outperformed the heuristic method. The importance of the constraint-based approach was also demonstrated by J. Hoffmann *et al*. (2024). When developing a heuristic algorithm, the authors verified its correctness and efficiency based on the integer linear model. The development by D. Briskorn *et al*. (2024) demonstrated the possibility of adapting models to highly specialised tasks. The flexibility of the approach made it possible to build a model of reorganisation for a specific enterprise, which, when using heuristics or other specialised approaches, would require the development of a unique algorithm. S.I. Bondariev (2020) investigated a mathematical model for planning transport processes in international road transport, in particular for calculating fuel requirements. However, the accuracy of the integer optimisation approach entails consequences in the form of long calculation times. This was demonstrated by J. Wessén *et al*. (2023) on medium-scale planning problems.

IBM, Google, Gurobi, and MiniZink offer a wide selection of open source packages for modelling and solving problems, with significant investments made in algorithm optimisation. The packages include the most promising algorithms, which has attracted a large number of application-level developers and researchers to use and improve them. The packages also include the latest hybrid approaches. The improvement to the CP-SAT algorithm proposed by A. Bit-Monnot (2023) has expanded the OR-Tools package and improved performance for resource-constrained planning problems. The author has developed a new strategy for eliminating conflicts in constraints when selecting variable values, which consists of quickly substituting the variable value throughout the chain of dependent equations and inequalities. In turn, F. Ulrich-Oltean *et al*. (2023) integrated machine learning for model coding into the CP-SAT algorithm. The developed classifier identified typical features of different classes of optimisation problems, which allowed for their optimal encoding. Thanks to optimal encoding, a reduction in computational complexity was achieved for the studied classes of problems. Hybrids of exact methods and machine learning, thanks to mutual compensation of weaknesses, showed better results than each of the methods separately.

Considering the above, the aim of the work was to develop an effective mathematical model that would allow successfully solving production order planning problems to optimise their execution time. It had to take into account the specifics of resources and the production process, as well as the correspondence between tasks and orders. The practical application of the system based on it will depend on the quality of the developed model.

## Materials and Methods

The mathematical model of order distribution formulated in this study is a formalised description of the planning problem, taking into account constraints, resource specifics, and dependencies between tasks and orders, expressed through a system of equations and inequalities with discrete variables. The integer model ensured accurate consideration of the discrete nature of the production process through the use of Boolean variables to model the assignment of tasks to resources and integer variables to reflect the start and end times of operations. The resource c in the production order planning problem is heterogeneous production equipment that performs only a certain type of task and may also have different productivity. The production order is specified by a directed acyclic graph $G_i$ and determines the sequence of task execution. The vertex of the graph $V_i$ determines the need for a certain type of resource, as well as the amount of time required for its execution. Tasks are performed strictly in the order specified by the graph $G_i$.

The solution to the problem included:

1. Establishing a correspondence between each task and resource in the form of a Boolean variable $a_{jc}$ value, where the value 1 corresponds to task $j$ being assigned to resource $c$.

2. Creating a schedule with task execution intervals. An interval is a pair of variable values $s_j$ and $f_j$, where $s_j$ is the start time of the task and $f_j$ is the end time.

The development of the mathematical model began with the formation of a system of constraints. The constraints were formed in accordance with the requirements of the physical production process. The first step was to establish a relationship between the task execution intervals and the time of completion of the entire order. The planned order execution time $f_j$ is defined as the time of execution of the last operation belonging to this order:

$$F_i = max\ \{f_j\}\ \forall i, \tag{1}$$

where $f_j$ – the time of completion of the operation, $i$ – the order index.

The planned start time $S_i$ of an order is determined as the start time of the first operation of that order:

$$S_i = min\ \{s_j\}\ \forall i, \tag{2}$$

where $s_j$ – start time of the operation, $i$ – the order index.

Each task is a time interval, the duration of which is determined by the input data length $d_j$, the task completion time is equal to its start time plus the execution time:

$$f_j = s_j + d_j\ \forall j, \tag{3}$$

where $s_j$ – task start time, $f_j$ – task completion time, $j$ – task index.

The tasks belonging to the order are performed strictly in the specified order. For each order $I \in [0, N]$, a graph $G_i = (V_i, E_i)$ is specified, where $V_i$ is a set of vertices representing the tasks of the order that must be performed, and $E_i$ is a set of edges specifying the order of their execution in accordance with the technology. The number of tasks in order $i$ is taken as $n_i$. Thus, the order of execution of tasks in order i is specified as follows:

$$s_j >= f_k\ \forall (j, k) \in E_i, \tag{4}$$

where $s_j$ – start time of the next task, $f_k$ – completion time of the previous task, $k$ – index of the previous task, $i$ – index of the next task, $E_i$ – set of connections between vertices.

For each resource from $c \in [0, C]$ at a given moment in time, only one task can be performed. This means that the task execution intervals of a single resource must not overlap. The set of tasks to be performed on a resource c is denoted by $J_c$. For each pair of tasks $(j, k) \in J_c$, the following must be performed:

$$s_j >= f_k\ or\ s_k >= f_j\ \forall (j, k) \in J_c, \tag{5}$$

where $s_j$ – task start time $j$, $f_j$ – task completion time $j$, $s_k$ – start time of task $k$, $f_k$ – end time of task $k$, $(k, i)$ – indices of a pair of operations.

To describe the constraint in linear form for each pair of tasks, a binary variable $y_{jk} \in [0, 1]$ is additionally introduced, which determines whether task $j$ is performed by the resource earlier than task $k$ (order of execution). Using the Big M method, it obtains a system of inequalities:

$$\begin{cases} f_j \le s_k + M * (1 - y_{jk}) \\ f_k \le s_j + M * y_{jk} \end{cases} \forall (j, k) \in J_c, \tag{6}$$

where $M$ – an infinitely large number, $y_{jk}$ – a binary variable that determines the order of operations.

Each task can be performed on only one resource. To denote the allocation of task $j$ to resource $c$, a binary variable $a_{jc} \in [0, 1]$ is introduced. All assignments are expressed by a matrix:

$$A = \begin{pmatrix} a_{00} & \cdots & a_{0c} \\ \vdots & \ddots & \vdots \\ a_{j0} & \cdots & a_{jc} \end{pmatrix}. \tag{7}$$

The time required to complete task $j$ using resource $c$ is specified in the input in the form of a matrix:

$$D = \begin{pmatrix} d_{00} & \cdots & d_{0c} \\ \vdots & \ddots & \vdots \\ d_{j0} & \cdots & d_{jc} \end{pmatrix}. \tag{8}$$

Then, the execution time of task $d_j$ can be expressed based on the assignment $a_{jc}$ and the input data matrix D:

$$\begin{cases} d_j = d_{j1}, \text{if } a_{j1} = 1 \\ \vdots \\ d_j = d_{jc}, \text{if } a_{jc} = 1 \end{cases}, \tag{9}$$

where $d_{jc}$ – is the time required by the resource c to perform the operation $j$.

In linear form, the constraint on the task execution time is expressed as follows:

$$d_j = \sum_{c \in C} d_{jc} * a_{jc}\ \forall j \in J_c, c \in C. \tag{10}$$

Since a single task can be assigned to only one resource, it is necessary to impose restrictions on the value of $a_{jc}$ so that in each row of matrix $A$ only one variable takes the value $a_{jc} = 1$:

$$\sum_{c \in C} a_{jc} = 1\ \forall j \in J_c, c \in C. \tag{11}$$

This ensures that the execution time of task $d_j$ corresponds to the value $d_{jc}$ specified for this resource by the input data matrix $D$.

The final stage of model synthesis was the development of an optimisation criterion. Minimising the completion time of the last order $O_L$ allows reducing the total

production time, and it is proposed to use this indicator as an optimisation criterion:

$$min \rightarrow O_L = max(F_i). \quad (12)$$

The designation INT_LIN_MK was introduced for this model, and further comparisons were based on this model as the baseline. The search for a solution to the model in the experiments was carried out using the Google OR-Tools package version 9.11 with default parameter settings. The implementation is based on the CP-SAT algorithm, which uses the logical inference method (SAT) as well as conflict-driven search, which ensures high performance when solving combinatorial problems.

## Results and Discussion

During the experiment, the dynamics of changes in calculation time depending on the number of orders and the associated number of variables in the model were analysed. Each order graph included 10 tasks, which corresponds to a typical configuration of the technological process. To study the speed of solving the problem using the proposed model, computational experiments were carried out with the number of orders ranging from 5 to 40 in increments of 5. Table 1 shows the data on the number of variables, the calculated value of the optimisation criterion and the calculation time for each of the experiments, which allowed a quantitative assessment of the scalability of the model.

A linear increase in the number of variables in the model was observed with an increase in the number of orders. At the same time, the increase in computation time was exponential – from 0.236 seconds for 5 orders to 2018 seconds for 35 orders – which demonstrated the difficulties of applying the model to large-scale problems. It was established that the maximum permissible dimension of input data for finding a solution using the basic model is limited to N = 35 orders. With 40 orders, no solution was found at all. To solve large-scale problems, it is necessary to consider a scaling method that will eliminate the exponential growth in calculation time (Gao *et al.*, 2019).

**Table 1.** Results of the INT_LIN_MK model

| No. | N | $O_L$ | Number of variables | Time, seconds |
|---|---|---|---|---|
| 1 | 5 | 691 | 381 | 0.236 |
| 2 | 10 | 1,186 | 691 | 2.5 |
| 3 | 15 | 1,681 | 1,036 | 14 |
| 4 | 20 | 2,176 | 1,381 | 110 |
| 5 | 25 | 2,671 | 1,724 | 345 |
| 6 | 30 | 3,166 | 2,071 | 774 |
| 7 | 35 | 3,661 | 2,416 | 2,018 |
| 8 | 40 | Not found | 2,856 | Not found |

**Note:** N – number of orders; $O_L$ – time of completion of the last order
**Source:** developed by the authors based on research

The key reason for the increase in calculation time is the growth in the number of variables. Finding the values of integer variables is an NP-hard problem (Camisa *et al.*, 2020), so the algorithm that performs the calculations of variable values has exponential complexity, which depends on the number of variables. To solve this problem, the INT_LIN_BATCH_MK model is proposed, based on the principle of sequential iterative application of the INT_LIN_MK model for a limited set of task variables.

To implement this approach, the SLICE_SIZE parameter is introduced, which limits the number of orders that are transferred in one iteration of distribution by the base INT_LIN_MK model. After the distribution of a subset of orders, the values of the variables $a_{jc}$, $s_j$, $f_j$ defined as optimal are recorded in a separate structure as an intermediate solution. In the next iteration, previously undistributed orders in the amount of SLICE_SIZE are included in the distribution. To comply with the rule that tasks performed on the same resource do not overlap, in expression (6) the intervals of tasks distributed in previous iterations are taken into account as constants. The proposed scheme is shown in Figure 1.

The application of the described approach made it possible to limit the number of variables whose values need to be defined at each iteration. Thus, it was possible to control the time spent on solving each iteration using the SLICE_SIZE parameter. To verify the reliability and evaluate the effectiveness of the model, a series of computational experiments was conducted. The experiments were conducted with the SLICE_SIZE parameter set to 5. A range of input data from 5 to 40 orders was used to correctly compare the results with the baseline model. Larger data sets ranging from 45 to 60 orders were used to analyse scalability and evaluate the performance of the model as its dimension increased. The results of the experiments are presented in Table 2 below.

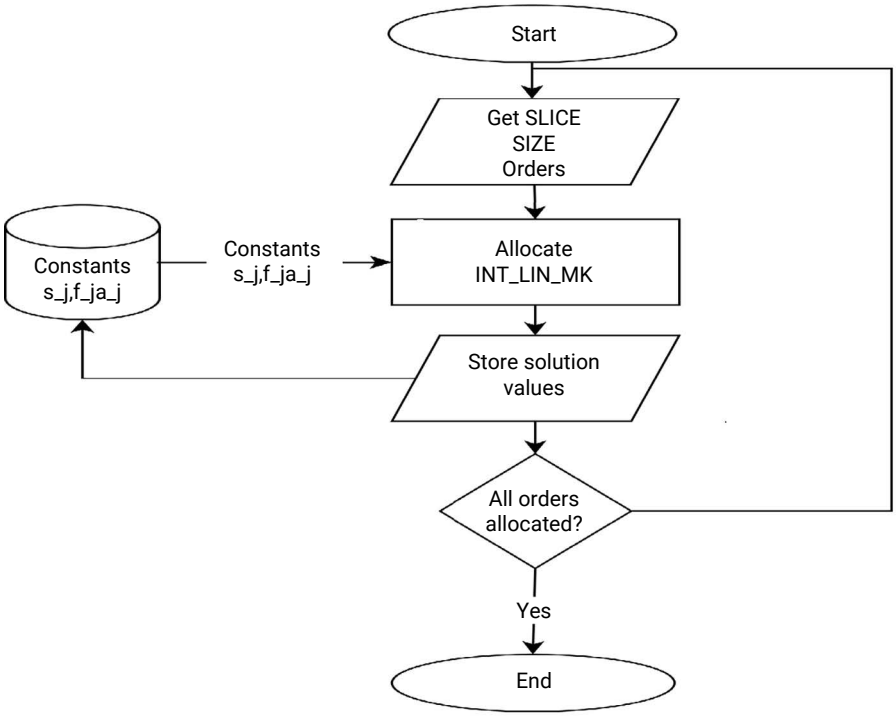



**Figure 1.** Block diagram INT_LIN_BATCH_MK
**Source:** developed by the authors based on research

**Table 2.** Results of the INT_LIN_BATCH_MK model

| No. | N | $O_L$ | Number of variables | Time, seconds |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

**Note:** N – number of orders; $O_L$ – time of completion of the last order
**Source:** developed by the authors based on research

Measurements show that there is a reduction in the number of variables for all experiments starting from the second, and most importantly, a reduction in the time spent on order distribution. This indicates that distributing orders in parts according to the INT_LIN_BATCH_MK model requires less time than distributing all orders at once. This phenomenon is explained by the exponential complexity of the basic CP-SAT algorithm and is a vivid demonstration of the divide and conquer principle applied by T. Tomesh *et al*. (2023) to other NP-hard problems. Based on the results obtained, it can be concluded that the improved model opens up opportunities for the practical application of integer linear programming and constraint programming approaches in resource allocation problems. The improved model made it possible to find solutions for dimensions that were not accessible to this approach and were solved in practice exclusively by greedy algorithms that provide results far from optimal (Lunardi *et al*., 2021). The SLICE_SIZE parameter allows balancing between the performance and accuracy of the solution, and depending on the need to increase it will allow obtaining a more accurate solution, while decreasing it will provide greater speed. For example, when SLICE_SIZE = 1, the model becomes equivalent to a greedy algorithm (Jiang *et al*., 2024).

When plotting the growth of the time required to distribute a given number of orders, a change in the shape of the curve to a more linear one is noticeable. This visualises the better scalability of the INT_LIN_BATCH_MK approach in terms of computation time and its wide range of capabilities for solving large-scale problems. An important achievement is the ability to estimate in advance the approximate time that will be required to solve the problem depending on the dimension (Fig. 2).

In addition to calculation time, memory consumption is an important aspect that affects the applicability of models for this type of task (Yang *et al.*, 2023). This is often overlooked during the development stage, leading to problems with system deployment and maintenance, especially in cloud environments, where it is not always possible to find capacity with the right capabilities at an acceptable price. In order to study the impact of the proposed method on this application aspect, an extended computational experiment was conducted using memory profiling with the memory-profiler package. Figure 3 shows a graph of memory usage dynamics when running the INT_LIN_MK (Fig. 3a) and INT_LIN_BATCH_MK (Fig. 3b) models on the example of N = 25 orders.

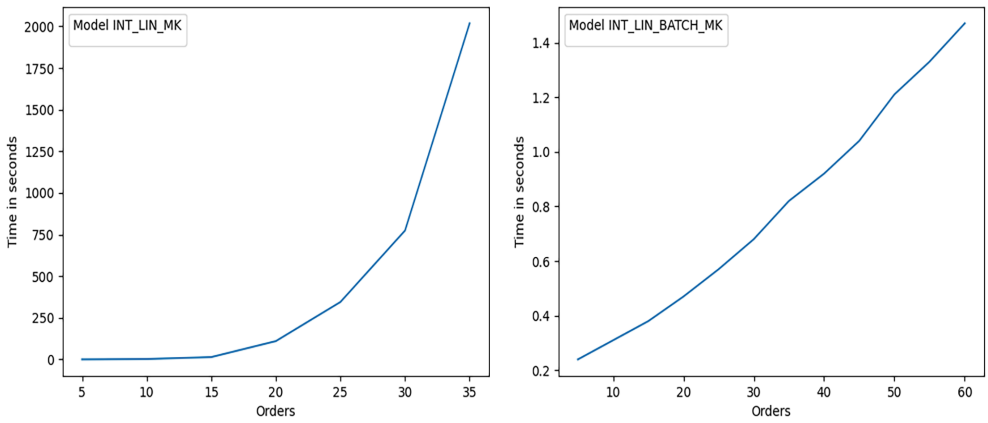
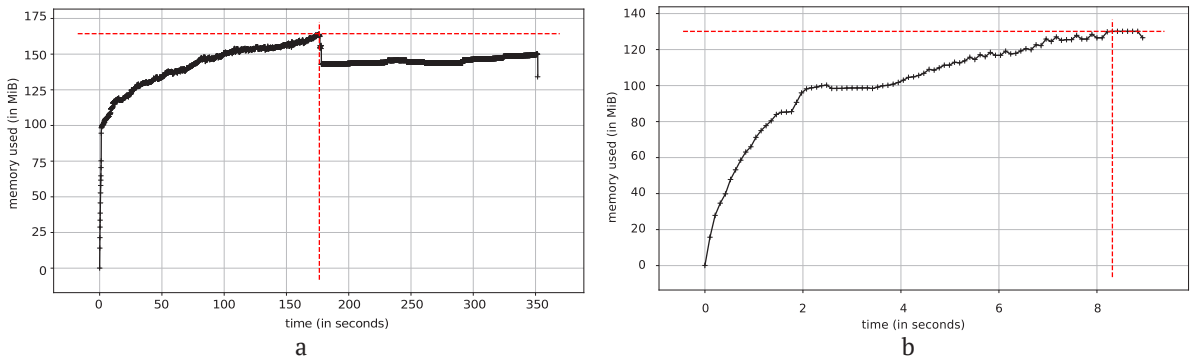



**Figure 2.** Comparison of model time consumption
**Source:** developed by the authors based on research




a b

**Figure 3.** Comparison of memory consumption over time
**Source:** developed by the authors based on research

Memory profiling results show that peak consumption reaches 182 MB for the base model and 134 MB for the improved model. This resulted in a 26% reduction in memory consumption. Although this seems insignificant for the volumes studied, the reduction in calculation time achieved allows the model to be applied to large input data sets. This, in turn, leads to an increase in the amount of memory used, so the consumption trend must be considered a key factor. Similar experiments were conducted for all available input data sets, the same ones used to test performance (Table 3).

**Table 3.** Comparison of memory consumption

| Model | N | Memory consumption, MB. | Memory consumption reduction, % |
|---|---|---|---|
| INT_LIN_MK | 5 | 127 | 0 |
| INT_LIN_BATCH_MK | 5 | 127 | 0 |
| INT_LIN_MK | 10 | 143 | 0 |
| INT_LIN_BATCH_MK | 10 | 127 | 11.2 |
| INT_LIN_MK | 15 | 151 | 0 |

Table 3. Continued

| Model | N | Memory consumption, MB | Memory consumption reduction, % |
|---|---|---|---|
| INT_LIN_BATCH_MK | 15 | 127 | 15.89 |
| INT_LIN_MK | 20 | 174 | 0 |
| INT_LIN_BATCH_MK | 20 | 130 | 25.29 |
| INT_LIN_MK | 25 | 182 | 0 |
| INT_LIN_BATCH_MK | 25 | 134 | 26.37 |
| INT_LIN_MK | 30 | 203 | 0 |
| INT_LIN_BATCH_MK | 30 | 137 | 31.03 |
| INT_LIN_MK | 35 | 234. | 0 |
| INT_LIN_BATCH_MK | 35 | 149 | 36.32 |

**Note:** N – number of orders
**Source:** developed by the authors based on research

The results of the measurements shown in Table 3 demonstrate that the rate of memory consumption growth for the improved model is significantly lower than the rate of cost growth for the baseline model. Additionally, the column "Memory consumption reduction" shows the ratio between memory savings and the baseline consumption determined by the INT_LIN_MK model for input data of this dimension. The percentage of savings shows a trend towards more efficient memory usage for large N. This proves the effectiveness of the proposed approach to model scaling and lower spatial complexity compared to the base model.

Below, in Figure 4, graphs of memory consumption growth for the corresponding models are shown. Similar to time costs, the proposed approach clearly demonstrates the ability to scale in dimension of memory and, in the future, allows calculations to be performed for large input data dimensions and is suitable for use in BigData systems. The base model could not be effectively applied for these purposes due to limited scalability capabilities.

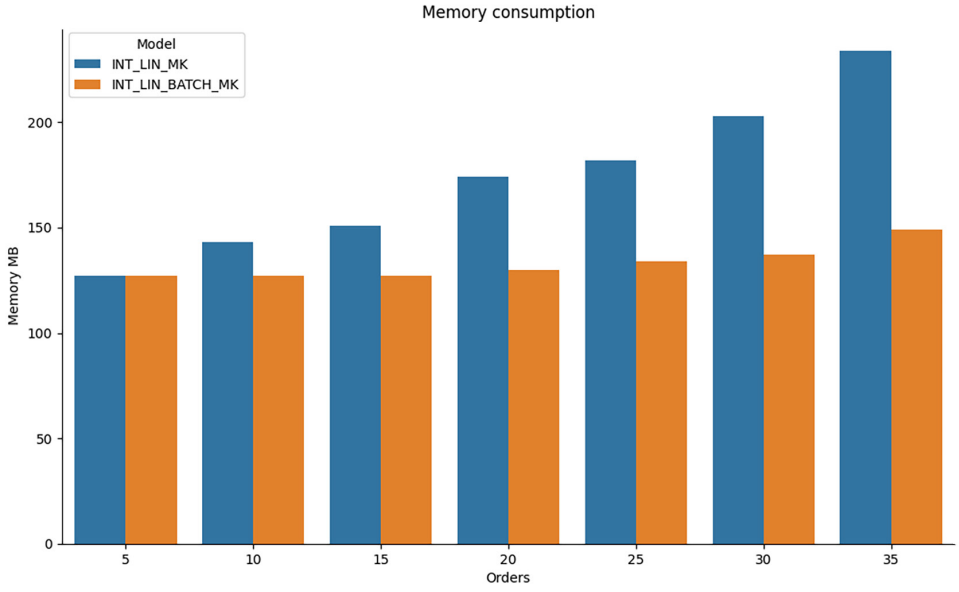



**Figure 4.** Comparison of memory consumption
**Source:** developed by the authors based on research

Empirical evidence confirms that INT_LIN_BATCH_MK demonstrates significantly higher performance and scalability compared to the base model. The experiments showed not only a linear grows in calculation time as the number of orders increased, but also a significant reduction in RAM (Random Access Memory) usage – up to 36.32% compared to the base implementation. This confirms the effectiveness of the applied strategy of solving the problem in stages using the SLICE_SIZE parameter, which provides control over the computational load and adaptability of the model to different input data dimensions.

The results obtained in the course of the research confirmed the importance of using hybrid models that combine the advantages of accurate and heuristic methods. Similar results were obtained by E. Guzman *et al*. (2021), who applied the grouping of several planned tasks into one with a combined execution time. The approach described by the researchers made it possible to reduce the dimension of the

task by applying a heuristic algorithm for grouping input data. This solution made it possible to achieve maximum resource utilisation, but made it impossible to optimise order delivery times. Unlike the model proposed in the current study, the algorithm operates on tasks rather than orders. In conditions of high production variability, the task grouping method showed poor results due to the impossibility of combining different types of tasks into one. This led to a decrease in productivity due to the high dimension of the input data. In addition, the INT_LIN_BATCH_MK model gained an advantage over the task grouping method by meeting order delivery deadlines, allowing orders to be planned as a whole without combining them with others.

In addition to decomposition by tasks and orders, there are also methods of decomposition by resources to which the distribution occurs. The algorithm for multi-level decomposition of the planning task by requested resources was developed by L. Wang *et al*. (2023) based on a classifier. The classifier allowed the formation of resource-independent subtasks based on a number of characteristics. Unlike the classifier, a parameter for precise control of the size of subtasks was developed for the proposed INT_LIN_BATCH_MK model. The introduced parameter of the number of orders for distribution – SLICE_SIZE – allowed for a precise breakdown into subtasks of the required size. In the case of SLICE_SIZE = 5, it was possible to achieve the expected time quantum of ≈ 0.11 seconds for the subtask, as shown in Table 2. The importance of this aspect for cloud systems is emphasised by H. Hozhabr Pour *et al*. (2025). The authors analysed the performance of the cloud platform depending on the size of the input data packet. This allowed to establish that increasing the packet size increases the throughput of the system, but also increases the total processing time. Adjusting the packet size parameter made it possible to configure the required throughput and performance in accordance with the system requirements.

An alternative algorithm that improves the performance of the planning problem solution was proposed by T. Azad *et al*. (2022). Similar to INT_LIN_BATCH_MK, an integer mathematical model was first synthesised. The researchers formed heterogeneous sets of input data for computational experiments. The input data size ranged from 20 to 200 orders and from 5 to 20 resources. For input data with a size of 20 orders and 10 production resources, the GA algorithm obtained a result close to the optimal one with a calculation time of 64 to 87 seconds. For the INT_LIN_MK model in Table 1, a calculation time of 110 seconds was obtained for a similar dimension. Thus, GA outperformed INT_LIN_MK by 20.9%-41.8%, which was the motivation for developing the improved INT_LIN_BATCH_MK model. Comparing Table 2 and the results of the GA algorithm provided in the study by M. Azad *et al*., it was found that the improved INT_LIN_BATCH_MK model outperforms GA. For an average dimension of 50 orders and 5 resources, INT_LIN_BATCH_MK provided a solution in 1.21 seconds, while GA took 169.96 seconds. Thus, INT_LIN_BATCH_MK showed 169 times better performance than GA on this set of input data. At the same time, INT_LIN_BATCH_MK and GA showed similar linear growth in calculation time, 2.16 and 2.5 times, respectively, when the input data size was doubled. Meanwhile, the base model INT_LIN_MK did not complete the calculation for a size of 40 orders.

Similar results also follow from experiments conducted by M. Heydar *et al*. (2021). The publication shows that an integer model similar to the basic INT_LIN_MK demonstrated low performance. For 100 orders, the calculation time using the IBM CPLEX package was 16,863 seconds. Given these time costs, the authors proposed using adaptive dynamic programming (ADP) to reduce the calculation time. For the aforementioned dimension of 100 orders, ADP found a solution in 15.08 seconds, which is a 1,120-fold acceleration. The INT_LIN_BATCH_MK model proposed in this work demonstrated a comparable acceleration relative to the INT_LIN_MK model. According to Table 1 and Table 2, for 35 orders, the calculation time decreased from 2,018 to 0.82 seconds, which means a 2,461-fold acceleration.

The next aspect of performance evaluation, but no less important than the calculation time of the algorithm, is memory consumption. Memory consumption is the factor that determines the infrastructure requirements for the system deployment. This is most relevant for cloud systems, as reducing infrastructure requirements has a direct financial impact. Y. Feng *et al*. (2020) investigated the financial losses caused by excessive memory reservation. According to the results, the reason for overspending is a lack of understanding of the system's actual memory needs and how to scale effectively. By relying on an improved scaling mechanism, the researchers managed to reduce memory consumption by 40%, which resulted in a 31% savings in cloud infrastructure costs. According to the data in Table 3, the INT_LIN_BATCH_MK model proposed in the current study provided memory savings of 36.32%, which, when applying the deployment method proposed by Y. Feng *et al*. and the calculations provided by the authors, gives an expected cost savings of 22%. Given the results obtained, the proposed model has become a valuable tool for order planning systems that are deployed in cloud environments and process a large number of customer calculations simultaneously.

## Conclusions

As part of the study, a scalable mathematical model was developed for the problem of planning production orders, taking into account limited production resources. The resulting integer model INT_LIN_BATCH_MK takes into account the rules and restrictions formed by the requirements of real production. The model is focused on large-scale tasks, which corresponds to the stated goal of the study – to achieve an effective solution to the task of planning large volumes of orders within an acceptable time frame, as confirmed by the results of the experiments conducted. Verification has demonstrated the ability of the proposed final model to ensure scalability and performance of calculations as the number of orders increases.

First, the INT_LIN_MK model was developed within the framework of the study, which implemented a complete formalisation of the planning task, including a strict order of execution of tasks included in one order, as well as the inadmissibility of simultaneous use of one resource for several tasks. The basic model ensured the finding of a global optimum, but its performance allows practical application only for low-dimensional tasks – up to 15 orders. Further growth in the dimension of the task led to an exponential increase in calculation time and a significant increase in memory usage. To expand the capabilities, a scalable model INT_LIN_BATCH_MK was developed, which provides for a step-by-step calculation of the plan on subsets of orders of a limited size, determined by the SLICE_SIZE parameter. An iterative combination of partial solutions was applied, with the values of the start and end variables of the tasks from the previous stages fixed as constants in the constraints, which ensures the consistency and integrity of the final schedule. This approach made it possible to scale the task to a larger number of orders without a significant increase in computational costs and ensured efficient processing of dimensions that were not available to the initial model. According to the results of the experiments, solving a problem with a dimension of 35 orders achieved a 2,461-fold acceleration in calculations compared to the base model, as well as a 36.32% reduction in operating memory consumption.

The results of the study confirmed that classical integer programming methods can be successfully adapted to large-scale problems using decomposition, as demonstrated by the iterative model INT_LIN_BATCH_MK. This opens up opportunities for expanding the scope of integer programming in applied planning systems, especially those deployed in a cloud environment, where scalability and calculation time are critical factors. Further research plans to examine in detail the impact of the SLICE_SIZE parameter on the quality of the calculated plan, in particular the degree of deviation from the global optimum, as well as to develop an algorithm for finding independent subsets of input data (no competition for a single resource) with parallel solution of each and subsequent combination of the results into a single order execution plan.

## Acknowledgements

None.

## Funding

The study was not funded.

## Conflict of Interest

None.

## References

[1] Andrade-Pineda, J.L., Canca, D., Gonzalez-R, P.L., & Calle, M. (2020). Scheduling a dual-resource flexible job shop with makespan and due date-related criteria. *Annals of Operations Research*, 291, 5-35. doi: 10.1007/s10479-019-03196-0.

[2] Azad, T., Rahman, H.F., Chakrabortty, R.K., & Ryan, M.J. (2022). Optimization of integrated production scheduling and vehicle routing problem with batch delivery to multiple customers in supply chain. *Memetic Computing*, 14, 355-376. doi: 10.1007/s12293-022-00372-x.

[3] Bit-Monnot, A. (2023). Enhancing hybrid CP-SAT search for disjunctive scheduling. In *26th European conference on artificial intelligence (ECAI 2023)* (article number hal-04174800). Krakow: PSSI.

[4] Bondariev, S.I. (2020). Optimization methods of operating costs on road transport in international transportation. *Machinery & Energetics*, 11(3), 129-133.

[5] Briskorn, D., Boysen, N., & Zey, L. (2024). Scheduling of e-commerce packaging machines: Blocking machines and their impact on the performance – waste tradeoff. *Journal of Scheduling,* 28, 101-120. doi: 10.1007/s10951-024-00826-9.

[6] Camisa, A., Notarnicola, I., & Notarstefano, G. (2020). Distributed primal decomposition for large-scale MILPs. *ArXiv*. doi: 10.48550/arXiv.2010.14446.

[7] Çetïnkaya, F., Yeloğlu, P., & Akkocaoğlu Çatmakaş, H. (2021). Customer order scheduling with job-based processing on a single-machine to minimize the total completion time. *International Journal of Industrial Engineering Computations*, 12, 273-292. doi: 10.5267/j.ijiec.2021.3.001.

[8] Feng, Y., Yan, F., & Luo, J. (2020). Memory scaling of cloud-based big data systems: A hybrid approach. *IEEE Transactions on Big Data*, 8(5), 1259-1272. doi: 10.1109/TBDATA.2020.3035522.

[9] Gao, K., Yang, F., Zhou, M., Pan, Q., & Suganthan, P.N. (2019). Flexible job-shop rescheduling for new job insertion by using discrete Jaya algorithm. *IEEE Transactions on Cybernetics*, 49(5), 1944-1955. doi: 10.1109/TCYB.2018.2817240.

[10] Guzman, E., Andres, B., & Poler, R. (2021). Models and algorithms for production planning, scheduling and sequencing problems: A holistic framework and a systematic review. *Journal of Industrial Information Integration*, 27, article number 100287. doi: 0.1016/j.jii.2021.100287.

[11] Heydar, M., Mardaneh, E., & Loxton, R. (2021). Approximate dynamic programming for an energy-efficient parallel machine scheduling problem. *European Journal of Operational Research*, 302(1), 363-380. doi: 10.1016/j.ejor.2021.12.041.

[12] Hoffmann, J., Neufeld, J. S., & Buscher, U. (2024). Minimizing the earliness – tardiness for the customer order scheduling problem in a dedicated machine environment. *Journal of Scheduling*, 27, 525-543. doi: 10.1007/s10951-024-00814-z.

[13] Hozhabr Pour, H., Ciortuz, G., Lüers, A., & Fudickar, S. (2025). Performance analysis of a data stream processing system for online activity classification via wearable sensor data. In *Proceedings of the 18th international joint conference on biomedical engineering systems and technologies – HEALTHINF* (Vol. 2, pp. 571-578). Porto: SciTePress. doi: 10.5220/0013166100003911.

[14] Jiang, X., Chen, Y., Chen, X., & Zhang, H. (2024). An optimal batch size determination method for time series deep learning models. *Sustainability*, 16(14), article number 5936. doi: 10.3390/su16145936.

[15] Lunardi, W.T., Birgin, E.G., Ronconi, D.P., & Voos, H. (2021). Metaheuristics for the online printing shop scheduling problem. *European Journal of Operational Research*, 293(2), 419-441. doi: 10.1016/j.ejor.2020.12.021.

[16] Musiał, K., Balashov, A., Burduk, A., & Rysińska-Wojtasik, D. (2023). Solving manufacturing orders scheduling problem using annealing simulation. In J. Machado, F. Soares, J. Trojanowska, E. Ottaviano, P. Valášek, M. Reddy, E.A. Perondi & Y. Basova (Eds.), *Innovations in mechanical engineering II. ICIENG 2022. Lecture notes in mechanical engineering* (pp. 279-288). Cham: Springer. doi: 10.1007/978-3-031-09382-1_25.

[17] Ofoghi, B., Mak, V., & Yearwood, J. (2020). A knowledge representation approach to automated mathematical modelling. *ArXiv*. doi: 10.48550/arXiv.2011.06300.

[18] Tomesh, T., Saleem, Z.H., Perlin, M.A., Gokhale, P., Suchara, M., & Martonosi, M. (2023). Divide and conquer for combinatorial optimization and distributed quantum computation. In *2023 IEEE international conference on quantum computing and engineering* (QCE) (pp. 1-12). Bellevue: IEEE. doi: 10.1109/QCE57702.2023.00009.

[19] Ulrich-Oltean, F., Nightingale, P., & Walker, J.A. (2023). Learning to select SAT encodings for pseudo-Boolean and linear integer constraints. *Constraints*, 28, 397-426. doi: 10.1007/s10601-023-09364-1.

[20] Wang, L., Liu, H., Xia, M., Wang, Y., & Li, M. (2023). Research on a multilevel scheduling model for multi variety and variable batch production environments based on machine learning. *Frontiers in Energy Research*, 11. doi: 10.3389/fenrg.2023.1251335.

[21] Wessén, J., Carlsson, M., Schulte, C., & Syberfeldt, A. (2023). A constraint programming model for the scheduling and workspace layout design of a dual-arm multi-tool assembly robot. *Constraints*, 28, 71-104. doi: 10.1007/s10601-023-09345-4.

[22] Yang, H., Zhao, M., Yuan, L., Yu, Y., Li, Z., & Gu, M. (2023). Memory-efficient transformer-based network model for traveling salesman problem. *Neural Networks*, 161, 589-597. doi: 10.1016/j.neunet.2023.02.014.

[23] Zhang, M., Lu, Y., Hu, Y., Amaitik, N., & Xu, Y. (2022). Dynamic scheduling method for job-shop manufacturing systems by deep reinforcement learning with proximal policy optimization. *Sustainability*, 14, article number 5177. doi: 10.3390/su14095177.

# Масштабована математична модель розподілу замовлень на виробництво

**Костянтин Гріщенко**

Аспірант
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
03056, просп. Берестейський, 37, м. Київ, Україна
https://orcid.org/0009-0008-9251-0222

**Олексій Писарчук**

доктор технічних наук, професор
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
03056, просп. Берестейський, 37, м. Київ, Україна
https://orcid.org/0000-0001-5271-0248

**Анотація.** У статті запропонована математична модель оптимізаційної задачі планування виробничих замовлень за умов обмежених ресурсів. Зростання варіативності виробництва зумовлює потребу у високопродуктивних і ефективних алгоритмах та моделях, які здатні обробляти велику кількість замовлень і мають високу адаптивність до нових критеріїв, правил та факторів. Метою дослідження був синтез масштабованої математичної моделі, що враховує правила побудови розкладів і дає змогу розв'язувати задачі великої розмірності в прийнятних часових межах. У роботі запропоновано дискретну математичну модель, що включає систему обмежень та критерій оптимізації. Модель відповідає реальним вимогам виробництва, зокрема враховує порядок обробки задач у межах одного замовлення та унеможливлює одночасне виконання задач на одному ресурсі. Для пошуку рішень моделі застосовано алгоритм CP-SAT, що входить до пакету Google Or-Tools і який визнано одним із найбільш продуктивних алгоритмів для задач дискретної оптимізації. Для наборів вхідних даних розмірністю від 5 до 40 проведено вимірювання часових і просторових затрат на розв'язання створеної моделі та знаходження оптимального рішення. З метою підвищення масштабованості підходу розроблено ітеративну модель розподілу замовлень частинами контрольованого розміру. У ній розмір підзадачі визначається параметром, який обмежує кількість замовлень, що включаються до базової моделі, дозволяючи регулювати обчислювальну складність. Результати експериментів продемонстрували, що масштабована модель забезпечила ефективне розв'язання задач розмірністю 60 замовлень за прийнятний для такого роду систем час. Порівняння базової моделі та покращеної засвідчило зниження часових затрат, а також споживання пам'яті для задач великої розмірності. При цьому зафіксовано лінійне масштабування як за часом розрахунку, так і за споживанням пам'яті при зростанні кількості замовлень, що гарантує ефективне застосування цієї моделі також для більших розмірностей. Отриманий результат досліджень створив підґрунтя для практичного застосування розробленої масштабованої моделі в інформаційних системах планування виробництва на підприємствах із високим навантаженням та великими обсягами замовлень

**Ключові слова:** оптимізація; програмування обмеженнями; дискретне лінійне програмування; математична модель; розподіл ресурсів; планування