Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 2026, 20(2), 243–263
https://doi.org/10.12913/22998624/211446
ISSN 2299-8624, License CC-BY 4.0

Received: 2025.10.15
Accepted: 2025.07.29
Published: 2025.12.23

Schedule optimization of a resource-constrained shipbuilding project using the delays increment method

Klaudia Skibińska[1*], Remigiusz Iwańkowicz[2]

[1] Department of Technological Processes, Faculty of Economics and Transport Engineering, Maritime University of Szczecin, Waly Chrobrego 1-2, 70-500 Szczecin, Poland
[2] Chair of the Department of Technological Processes, Faculty of Economics and Transport Engineering, Maritime University of Szczecin, Waly Chrobrego 1-2, 70-500 Szczecin, Poland
* Corresponding author's e-mail: k.skibinska@pm.szczecin.pl

ABSTRACT

This article demonstrates that shipbuilding project planning under uncertainty and resource constraints can be effectively modeled as a resource-constrained project scheduling problem (RCPSP), as described in the literature. The study highlights the specific challenges of the shipbuilding industry and the necessity of developing an efficient optimization approach tailored to complex projects characterized by stochastic activity durations. The challenge lies not only in achieving computational efficiency but also in enabling practical use of the generated solutions within real-world decision-making processes. A parameterized optimization control method is proposed, incorporating a predefined level of risk acceptance. High computational efficiency was achieved through the development of a controlled simulation technique referred to as the delays increment method (DIM). This method employs timed Petri nets and so-called delay vectors, which enable a recursive search for increasingly optimal schedules. The proposed model operates exclusively within the space of feasible solutions, significantly accelerating the optimization process. The practical applicability of the proposed approach is illustrated using a real-world shipbuilding project. Furthermore, two different risk management strategies and their respective impacts were tested. The conclusions indicate potential directions for further development of the method and identify challenges associated with its implementation in industrial practice.

Keywords: resource-constrained project scheduling, petri net simulation, delays increment method, shipbuilding, random-time process.

INTRODUCTION

Shipbuilding represents one of the most complex sectors of industry, combining organizational, technological, and economic requirements. As a result of a multi-stage production process, a vessel is constructed – one of the largest and most structurally complex means of transportation manufactured today. Managing the hull construction process involves scheduling thousands of activities, synchronizing deliveries from subcontractors, and coordinating the work of numerous labor teams. Both external factors and internal conditions affect the course of production, increasing the risk of disruptions at various stages.

Effective management of shipyard production requires the use of specialized IT tools tailored to the complexity and one-of-a-kind nature of production processes [1]. In practice, ERP, MRP, and MES-class systems are implemented to integrate information across production processes. One of the approaches developed in this area is the concept of lean shipbuilding, proposed by Song and Zhou [2], which involves defining task packages corresponding to hull sections and managing them via MES systems. In line with the core assumptions of lean philosophy, the focus is on waste elimination and streamlining the flow of information. It is worth noting that the proposed system generates schedules based on a predefined

assembly plan and the availability of human and spatial resources, without applying optimization algorithms. Data collection and the ability to trace the virtual flow of production form the basis of the digital twin concept in the shipyard environment. According to Okamoto [3], the digitization of production activities and a system-wide approach that models the entire shipyard are essential. To fully leverage the potential of such a framework, it is necessary to extend the system with a module supporting real-time operational decision-making.

There is a justified need to develop a tool dedicated to the shipbuilding industry that enables dynamic generation and updating of production schedules. This paper presents a project-oriented scheduling approach for the hull section assembly process, aimed at capturing real-world constraints. First, a literature review is conducted on the resource-constrained project scheduling problem (RCPSP) and existing methods dedicated to ship hull production planning. Subsequently, a shipyard production model is developed, incorporating various types of resources, uncertainties in activity durations, and the potential occurrence of additional tasks arising from quality control. For the proposed model, a schedule risk management strategy and an optimization algorithm are formulated. The method is based on the sequential timed resource-constrained Petri net (STRCPN) and employs a schedule control mechanism by introducing activity delays. The effectiveness of the approach is validated using a test instance that simulates the assembly process of hull sections.

BACKGROUND

Resource-constrained project scheduling problem

Project scheduling under limited resource availability, known as the RCPSP, is one of the central issues in operations research and contemporary project management. The problem was formally defined by Blazewicz et al. [4], who classified its structure and demonstrated its computational complexity, thus initiating extensive research in this area. The classical RCPSP consists in determining a feasible execution schedule for a set of interdependent tasks, while simultaneously accounting for limited renewable resource availability, precedence relations, and the minimization of the total project duration (makespan) [5].

In recent years, numerous extensions to the classical RCPSP model have been proposed. These include multi-project scheduling [6,7], consideration of multi-skilled resources [8], allowance for task preemption [9], and the integration of sustainability aspects, such as energy consumption [10]. Other developments have included modeling resource transfer times between tasks [11] and multi-objective optimization approaches addressing cost, reliability, and workload balancing [12].

The stochastic, multi-mode RCPSP extends the classical model by incorporating two sources of uncertainty – variability in task durations and the existence of alternative execution modes. Task durations may be modeled as random variables with known probability distributions or as sets of discrete scenarios [13–16]. In many approaches, execution mode selection is treated as a decision made during preliminary optimization and is assigned to tasks before scheduling begins. Additionally, researchers have introduced alternative paths within the project graph [17, 18], enhancing structural flexibility and enabling the modeling of alternative execution variants – for example, performing a task manually or via automation.

A variety of methods have been proposed to solve RCPSP and its variants. For small-scale projects, exact methods such as integer programming and branch-and-bound algorithms are applied despite their high computational complexity [19]. For larger, more complex problems, metaheuristic approaches prevail, including genetic algorithms (GA), particle swarm optimization (PSO), simulated annealing (SA), ant colony optimization (ACO), and their hybrids, such as iterative improvement procedures [20].

To support optimization algorithms, simulation techniques have also been employed—particularly useful in dynamic and stochastic environments. Liu et al. [21] used Monte Carlo simulation to support a genetic algorithm in solving a stochastic, multi-mode RCPSP. In that study, simulation served to evaluate the quality of solutions generated by the GA. Satic et al. [22] modeled a dynamic and stochastic RCPSP as a discrete-time Markov decision process (DR-MDP). Simulation provided data for regression and facilitated training of a linear value function.

Among simulation tools, Petri nets are particularly notable for modeling dependencies and analyzing project execution. Their graph structure with tokens enables the tracking of network state

and naturally captures concurrency, precedence constraints, and resource allocation. A key advantage of Petri nets is that resource constraints are inherently satisfied – simulation does not permit their violation. This is a result of their fundamental firing mechanism: transition enablement depends on the number of tokens in preceding places [23]. If the place representing a given resource pool lacks sufficient tokens, the corresponding task (transition) cannot be initiated.

In the context of RCPSP, Petri nets are often used as standalone schedule generation mechanisms, which can then be optimized using suitable algorithms. Prashant Reddy et al. [24] applied Petri nets to model a multi-mode RCPSP with task preemption. Scheduling was based on a Petri net model combined with a genetic algorithm that searched for optimal mode combinations minimizing project duration or cost. Niño et al. [25] used Petri nets to represent a project within the classical RCPSP framework. Their approach utilized a beam search algorithm (BASPS) to explore the Petri net's state space and select the most time-efficient schedule. Hu and Wang [26] proposed an extended Petri net structure to address RCPSP in multi-project environments. Scheduling was performed through simulation, and transition rules reflected decisions about task initiation, delay, or blocking, depending on resource availability.

Scheduling of ship hull construction

In the shipbuilding industry, production is typically single-unit and project-based in nature. A ship's structure is divided into blocks, sections, and components required for assembly. At each stage of the process, a sequence of activities is performed, such as cutting steel plates and profiles, machining parts, welding components, quality control of welded joints, and transportation. These tasks must be executed in a prescribed technological and structural sequence, forming a precedence dependency structure that determines feasible production schedules. However, the execution of this sequence also requires the availability of specific resources, which play a critical role in shipyard production.

According to the classification proposed by Storch [27], shipyard resources can be divided into four categories: material, manpower, facilities, and expenses. In practice, each category includes a wide range of renewable and non-renewable resources received in scheduled deliveries. Their diversity is considerable—from various steel grades, paints, and prefabricated components (e.g., engines, pipes, cables), to skilled personnel and specialized transportation equipment and infrastructure.

The production of a single hull section can take several weeks, while the entire project may span many months. The long-term nature of the process necessitates flexible planning that accounts for potential disruptions. In shipbuilding, inspection and measurement activities play an important role in continuously evaluating assembly quality. These may trigger additional corrective tasks to fix defects, non-conformities, or deviations from the design. This impacts the overall project timeline, causing delays in subsequent stages and increasing resource consumption. Corrective actions require additional labor, which – given limited human resources – necessitates careful reallocation decisions.

The literature on ship hull construction focuses on production planning, scheduling, and estimation of labor hours. Okamoto and Hirakata [28] summarized prior work, emphasizing the importance of optimization algorithms (GA, ACO, tabu search) and modeling techniques such as Petri nets and discrete-event simulation in reflecting the specific nature of shipbuilding processes. Makoto and Nagaoka [29] applied a multi-objective genetic algorithm (MOGA) to optimize hull construction schedules, showing how workforce size influences project duration. Skibińska [30] used the FlexSim environment and Optimizer module for human resource allocation in hull section assembly. Li et al. [31] considered non-renewable resources, including prefabricated elements delivered during the project, and applied the improved Grey Wolf optimizer (IGWO) to minimize total project makespan.

Petri nets were used by Yu-guang et al. [32] to model and simulate the panel hull block assembly system (PHBAS). To represent different block types, they employed a colored, timed Petri net. Simulations of a sample process involving three hull block types yielded an optimal execution sequence maximizing productivity and resource utilization. Jeong et al. [33] proposed a method for modeling block assembly processes using Petri nets. Their model integrates products, processes, and resources, enabling formal analysis and simulation of existing schedules. The authors identified their model as a starting point for

future work, particularly in extending the method toward optimal schedule generation.

Existing research on RCPSP and its extensions highlights directions for the development of scheduling methods for complex and disruption-prone projects. This creates a foundation for models that more accurately reflect real-world shipyard processes. Although several studies have addressed hull section scheduling, they remain relatively scarce and often overlook the aspects of flexibility and uncertainty. The diversity of resources employed in shipyards constitutes a limiting factor for the application of classical RCPSP models. These models must be extended to account for both renewable resources (e.g., workers, equipment, workstations) and non-renewable resources (e.g., materials, prefabricated components). In addition, it is necessary to incorporate various supply modes, since deliveries may occur either cyclically or as single shipments at predefined dates.

A further practical challenge lies in the large scale and complexity of shipyard production. In such cases, the key issue is not merely to find an optimal solution as quickly as possible, but rather to identify any feasible and acceptable schedule at all.

Petri nets, previously used mainly as modeling tools, can – when appropriately extended – also support optimization. This article builds upon this direction by proposing a model tailored to the specifics of hull section assembly and an algorithm based on Petri nets with delay-driven schedule control.

MATERIALS AND METHODS

Assumptions

The study considers the production processes of large-scale welded structures with stiffened shell-type configurations. A structured decomposition of welded structures into technological assemblies has previously been proposed by Iwańkowicz [34], providing a formal foundation for planning and standardization in large-scale offshore and shipbuilding environments. The project is decomposed into a set of activities whose start times constitute the target schedule. The execution of each activity involves the consumption of specific renewable and non-renewable resources. The solution space is constrained by the predefined activity sequence and the availability of required resources. The selection of the optimal schedule is based on the criterion of minimizing the total project duration (makespan).

The proposed model utilizes a single-partite activity-on-node (AoN) graph to visualize the sequence of tasks and analyze the critical path, as well as a bipartite graph, represented as a Petri net, to simulate the project execution process.

The computational algorithm is based on the theory of directed acyclic graphs (DAGs) and their representation using nilpotent matrices.

All time-related analyses were carried out using dimensionless time units to ensure the generality of the study and the clarity of the results presented in the Gantt charts. The proposed methods and models are scalable with respect to the overall duration of the processes under investigation. In practice, a rational unit of time in the shipbuilding industry is either one hour or one working day. However, there is no limitation to adopting alternative units, such as several hours or several days. A general practical recommendation is to select the unit in such a way that the total process duration is expressed in the order of hundreds of units.

The project model incorporates stochastic execution modes for selected activities and random durations within each mode. In the examples presented, activity durations follow a normal distribution; however, the final choice of distribution is flexible and depends on the available statistical data.

The developed approach consists of two integrated components:

- a decision-making procedure, addressing the need for management under uncertainty and controlled risk-taking,
- and an optimization method based on delay increments, referred to as the delays increment method (DIM), which enables the generation of feasible schedules for given project scenarios.

The proposed optimization technique is supported by a simulation model implemented in MATLAB R2025a. The model is based on a timed Petri net, in which the transition firing time represents a delay in token generation relative to the moment of activation.

During the computations, the default Mersenne Twister random number generator in MATLAB was employed, with the seed automatically initialized based on the system clock.

Numerical experiments were conducted on a workstation equipped with an Intel i7 3.40 GHz processor and 32 GB of RAM.

Stochastic character of the model

The considered shipyard production processes exhibit variability in the scope of operations shaping the geometry of the welded structure. Depending on the extent of thermal deformations, dimensional quality control may indicate the need for different types of corrective actions.

It is assumed that the ship hull subassembly project consists of *n*probability of detecting dimensional imperfections activities that must be executed. These may include operations such as welding, grinding, dimensional control, straightening, or internal transportation. Activities may be interrelated by precedence constraints represented as directed arcs in the process graph. The project may also include control and correction (CC) activities, involving control of the welded structure to verify whether the dimensions fall within allowable tolerances.

The outcome of a control activity may either confirm dimensional compliance or lead to a recommendation for corrective actions. These may include additional heating, the application of external forces, or – under severe deviations – partial reconstruction of the structure. It is assumed that corrective procedures also involve auxiliary control operations integrated into the project workflow.

Control and correction activities are denoted as CC and are considered in two execution modes:

- Mode 1: Metrological control confirming compliance with tolerance limits,
- Mode 2: Detection of dimensional imperfections followed by corrective actions.

Let Q denote the subset of all control and correction activity indices, where $Q \subseteq \{1, 2,..., n\}$. For each activity $i \in Q$, the probability of detecting dimensional imperfections is denoted by $p_i \in [0,1]$.

For example, in a project comprising 8 activities (see Figure 1), assume that activities 2 and 5 are designated as control and correction tasks. In such a case, four execution variants are possible:

- Variant 1: Both controls confirm dimensional compliance.
- Variant 2: Corrective actions are required after activity 2.
- Variant 3: Corrective actions are required after activity 5.
- Variant 4: Corrective actions are required after both activities 2 and 5.

In the proposed model, activity durations are treated as random variables with probability distributions. In the case of repetitive tasks, the distribution of activity durations is determined on the basis of statistical analyses of real-world processes previously carried out within the given production system. For one-off processes, however, it is necessary for a technological expert to estimate the nature of the distribution. The simplest solution in such cases may be to assume certain boundary values and apply a uniform distribution.

Each CC activity has two alternative execution modes. A separate random distribution of duration is defined for each mode. The distribution used in simulation depends on the specific process scenario under consideration. The duration of the activity is shorter when the inspection result confirms compliance, compared to the case in which additional corrective actions are required.

Illustrative values of expected durations and variances for a set of eight activities corresponding

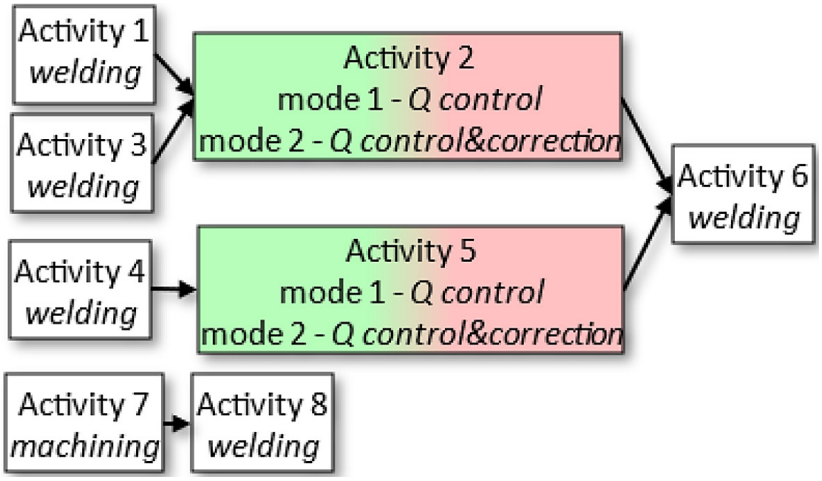


Figure 1. Alternative execution variants of the example project

to the process shown in Figure 1 are provided in Appendix A, Table A1.

Project resource consumption

Each activity in the project may consume certain resources. These can be renewable resources, such as labor, machine time, workspace, storage area, and transport means. These resources are occupied during task execution and become available again upon completion. If the project has access to a sufficient amount of a given resource, multiple activities can use it concurrently. It is assumed that renewable resource limits are predefined and remain constant throughout the time horizon considered in a single simulation run. A key criterion in project planning is to maximize the utilization of available renewable resources, i.e., to minimize idle time and underutilization.

The second category consists of non-renewable resources, including raw materials, prefabricated components, and energy. These are supplied to the facility and often stored in specific quantities to safeguard future processes. It is essential to minimize stock levels to avoid excessive storage costs and frozen working capital. Non-renewable resources are drawn from a warehouse, which has an initial stock level. Replenishment is possible via scheduled deliveries of fixed quantity and frequency.

If the project consumes r different types of resources, then each activity is characterized by a vector of r non-negative consumption values. For renewable resources, these values reduce the global pool for the duration of the activity. For non-renewable resources, they are subtracted from inventory at the moment the activity begins and are not replenished.

CC activities – implemented either as pure inspection or including corrective actions to restore geometry – differ significantly in resource consumption. Corrective operations typically require welders, blacksmiths, special equipment, and additional materials. Extra inspections are also necessary in this mode. Therefore, each such activity is assigned two alternative resource consumption vectors, corresponding to the two execution modes. The probability of selecting a given variant is consistent with the probability distribution assigned to the duration of the activity. Illustrative resource demands for each activity are provided in Appendix A, Table A2.

Resource availability and supply model

Resource availability is specified using a matrix $RA = (ra_{k,i})_{r\times 4}$, where:

- $ra_{k,1}$ – initial stock level of resource k at the beginning of the project,
- $ra_{k,2}$ – quantity delivered in a single replenishment of resource k,
- $ra_{k,3}$ – time of the first delivery from the process start, and the fixed interval between subsequent deliveries of resource k,
- $ra_{k,4}$ – total number of scheduled deliveries for resource k.

The pool of renewable resources decreases during task execution and is replenished immediately upon completion. The pool of non-renewable resources may be replenished through planned deliveries during the process. Each delivery is characterized by a fixed quantity and fixed interarrival time.

An example of a resource availability matrix for three resources is presented in Appendix A, Table A3. A graphical representation of time-phased resource availability, considering delivery timing and quantity, is provided in Figure A1.

Sequential timed resource-constrained Petri net

For a given project composed of activities with predefined precedence relations, a sequential timed resource-constrained Petri net (STRCPN) can be constructed. In this structure:

- Transitions T1, T2,..., Tn correspond to the individual project activities.
- Places model the precedence dependencies between activities.

Each transition Ti is assigned an activation time, corresponding to the duration of the activity. The activation time refers to the time delay between the consumption of input tokens and the generation of output tokens. While a transition is active, it does not consume further tokens. The end of activation results in generating tokens in subsequent places, thus enabling downstream transitions.

Activity durations are randomly sampled from probability distributions defined for the specific scenario. Particular attention is paid to transitions representing CC tasks. Their durations are sampled based on the distribution corresponding to the execution mode (inspection only or control + correction).

In addition to standard transitions and places, the network includes structures that model resource availability mechanisms.

Figure 2 illustrates an example STRCPN for the test project shown in Figure 1, assuming a scenario in which both CC activities (2 and 5) confirm compliance, and no corrective actions are needed. Transition indices 1 through 8 correspond directly to activity IDs. The example includes three types of resources:

- Resources 1 and 2 are renewable; their availability is modeled by token counts in places P13 and P14,
- Resource 3 is non-renewable; its inventory is represented by place P15. Replenishment occurs when transition T9 fires every 10 time units, increasing the resource pool by one unit. The number of possible deliveries is limited by the number of tokens in place P16.

The initial marking of the Petri net must account for tokens placed in the initial places that trigger the simulation. In the discussed example, these are places numbered 1, 3, 5, and 10.

Places representing the pools of renewable resources must initially contain tokens equal to the quantity of available resources (Figure 2, places 13 and 14). In turn, the place representing the non-renewable resource warehouse contains tokens corresponding to the currently held stock. The limited number of deliveries for non-renewable resources is modeled via tokens placed in an auxiliary place, which feeds transition T9 responsible for replenishing the warehouse P15.

For the presented example in Table 1, the initial marking of the net is as follows.

It is important to note that resource limitations can introduce stochastic behavior into the STRCPN model. In the scenario depicted in Figure 2, at time 0, activities 1, 3, 4, and 7 may simultaneously attempt to use resource 1 for activation. Whether these activities begin in parallel or only a subset of them can proceed depends on the availability of the resource pool. In such cases, the selection of which activities proceed is random, with each competing activity having equal probability of activation.

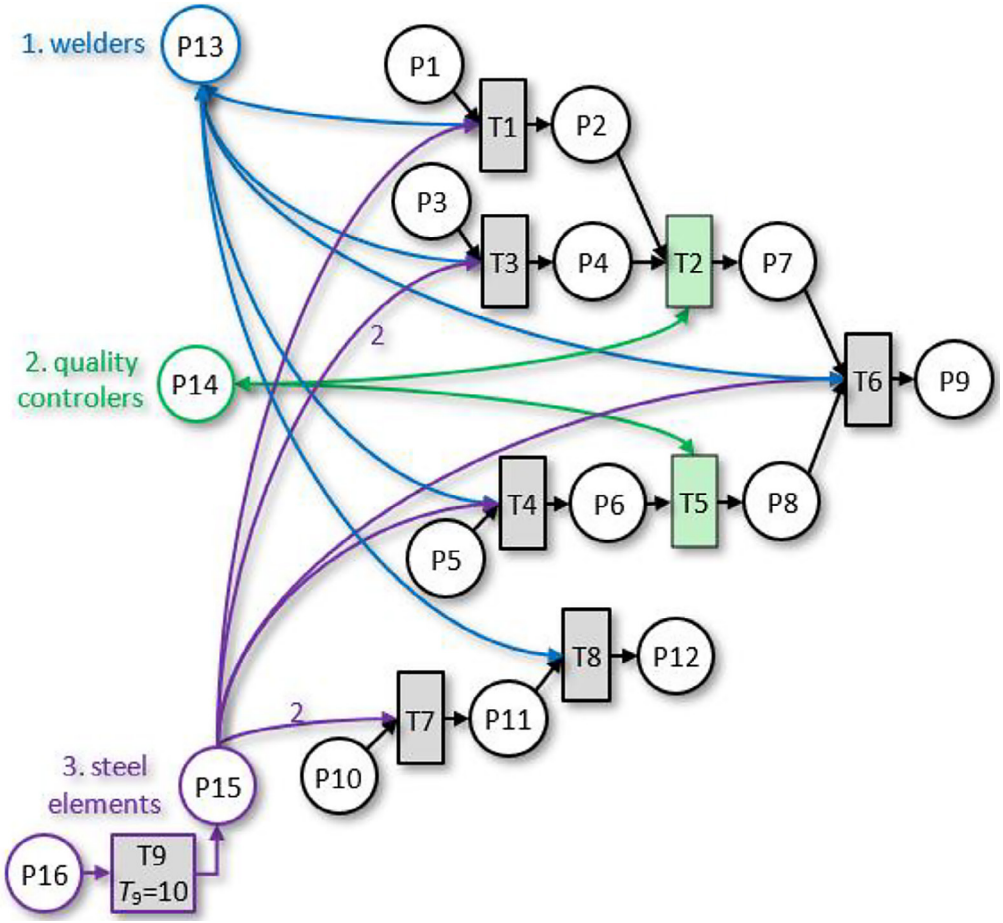


Figure 2. Example STRCPN – Scenario 1

Table 1. The initial marking of the net

Place:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Tokens:	1	0	1	0	1	0	0	0	0	1	0	0	2	1	3	2

As a result of the simulation, we obtain a sequence of network states that allows tracking of transition activation and completion times. Based on this data, it is possible to determine the start and end times of individual activities. The stochastic nature of activity durations is captured by performing multiple simulation runs. The resulting set of schedules is subjected to statistical analysis, enabling estimation of the probability of key events occurring within specified time windows.

Stochastic project realizations

The simulation study investigates a project in which CC activities are executed in randomly selected modes, and all activity durations are subject to stochastic variability. Consequently, the resource consumption associated with CC activities also exhibits randomness, as it depends on the selected execution mode.

Each stochastic realization of the project is recorded as:

- a vector of activity durations, and
- a corresponding resource consumption matrix.

For the process defined in Figure 1 and in Tables A1–A3 of Appendix A, 10 scenarios were randomly generated. These scenarios are listed in Appendix A, Table A4. Each scenario includes sampled durations for all activities. It can be observed that CC (activities 2 and 5) involve corrective actions in some of the scenarios.

For each randomly generated project scenario, a simulation can be performed using the Petri net model described in the previous section. Since the simulation procedure is stochastic, the resulting schedule represents a random realization of the process. To illustrate this phenomenon, 100 simulations were conducted for each of the 10 scenarios presented in Table A4 (Appendix A). Each set of simulations produced a collection of distinct schedules, even for the same scenario.

Figure 3 shows the results of three selected simulations for Scenario 1 from Table A4, highlighting the variability of activity start and end times arising from stochastic resource competition and duration distributions. Activities are shown in ascending order from bottom to top.

It was observed that both the progression and total duration of the project depend strongly on the selection of initial activities. Due to resource availability constraints, the project may begin with the following activity pairs: (1 and 3), (3 and 4), (1 and 4), (4 and 7), or (1 and 7).

In the schedule shown in Figure 3a, where the project starts with activities 4 and 7, the completion time was 49 time units. In contrast, the schedule in Figure 3b, which starts with activities 3 and 4, resulted in a reduced makespan of 40 time units. Thus, the choice of initial activities influences the overall schedule structure, although the further progression of the project also plays a significant role.

In Figure 3c, which also begins with activities 3 and 4, the shortest project duration was obtained – 35 time units. Despite having the same starting activities as in Figure 3b, the downstream sequence of events differed, ultimately impacting total project time. In particular, the availability of resource 3 at time 10 limits the possibility of executing activities 1 and 7 simultaneously. Simulation results show that initiating activity 1 and delaying activity 7 yields a better overall result.

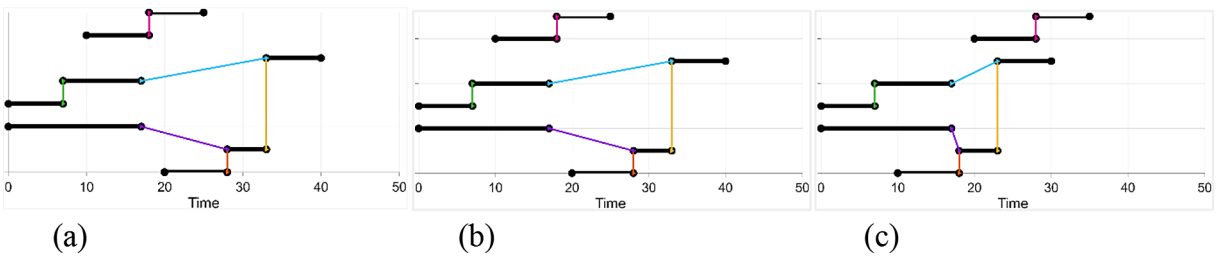


Figure 3. Comparison of Gantt charts for three realizations of scenario 1

The stochastic nature of project realization arises from the properties of Petri nets: during each simulation, transitions are randomly selected from the set of enabled transitions. The simulation results indicate that the order of activity execution decisions significantly affects schedule efficiency. Purposeful introduction of delays can be used as a schedule control mechanism, a topic further explored in next sections.

SCHEDULE OPTIMIZATION

Risk management strategy

Finding an optimal schedule under uncertainty requires defining an acceptable risk threshold for exceeding the planned project completion time due to potential delays. A schedule with overly strict start and finish times is highly prone to disruption – especially in the case of significant deformations in welded shipyard structures. A high frequency of control + correction tasks can severely delay execution of the planned workflow.

On the other hand, a schedule with excessively large time buffers is suboptimal with respect to the objective of minimizing the total project duration. In such cases, randomly realized task durations will statistically tend to complete earlier than their planned finish times, resulting in idle workers and underutilized resources.

Therefore, the goal is to identify a schedule that:

- minimizes project duration as much as possible,
- while simultaneously ensuring that the statistical probability of delay remains below a predefined risk acceptance threshold.

Schedule optimization is performed for a given project scenario, defined as a vector of activity durations and the corresponding resource consumption levels. The scenario is determined according to the following procedure:

- Generate a large sample of random project realizations based on assumed execution modes and distributions of activity durations. A sample size of at least 30 is required, though larger samples are recommended for improved accuracy.
- Calculate the mean μ_i (μ mean duration time of activity from random generated scenarios) and standard deviation σ_i for the duration of each activity i from the sample.
- Compute the scenario-specific duration for each activity using the formula:

$$T_i = \mu_i + k \cdot \sigma_i \tag{1}$$

where: $k \in [0,3]$ is a risk control coefficient, reflecting the level of conservatism accepted in the planning process.

For CC activities, the resource consumption is assumed in the pessimistic variant, corresponding to the execution mode with geometry correction.

Figure 4 presents the sequence of steps in preparing data for the schedule optimization process, the moment of incorporating the coefficient k into the decision-making process, and the information used for its verification.

The coefficient k is a decision parameter: the higher its value, the longer the task durations used as input to the DIM optimization procedure. These durations are treated as the planned activity times in the resulting schedule. Consequently, increasing the value of k results in longer planned project durations, offering higher schedule robustness at the cost of efficiency.

Simulation studies indicate that a maximum safe value of $k = 3$ yields a very low probability of exceeding the planned project deadline.

Setting $k = 0$ implies that the planned schedule is based on mean activity durations from the sample, which inherently carries a 50% probability of individual task delays. The impact of such delays on final project completion depends on the project's structure and resource constraints.

Based on the value of k, two representative meta-level scheduling strategies can be defined:

- $k = 1$: risk-seeking strategy,
- $k = 3$: risk-averse strategy.

In the section concerning the application of the DIM algorithm, a comparative example is presented to demonstrate the impact of adopting different scheduling strategies. The evaluation is based on the results of repeated project simulations under random realizations. From the resulting empirical distribution of project completion times, the probability of exceeding the planned deadline is estimated.

Delays increment method

Conventional optimization methods used in RCPSP problems are computationally inefficient when applied to the considered shipbuilding

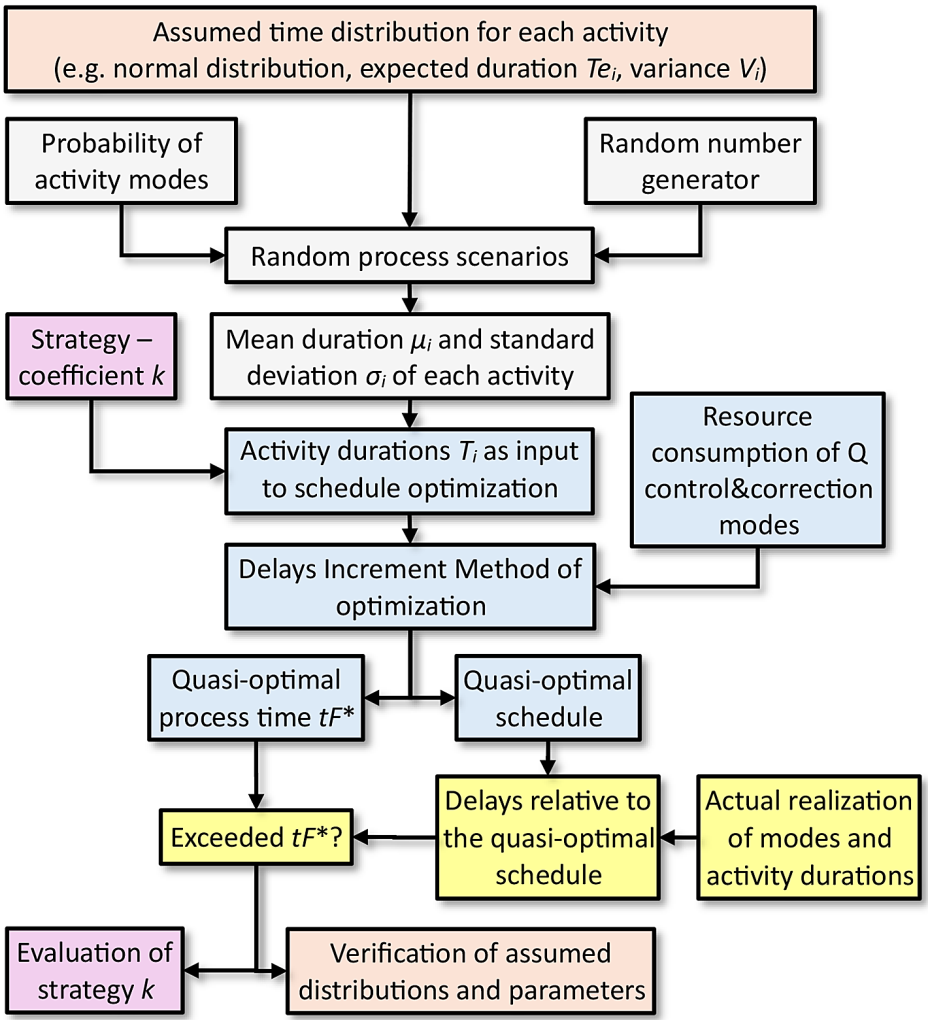


Figure 4. Information flow during data preparation for optimization and verification of the adopted assumptions

project instances. The main challenges include the large number of activities and the difficulty in identifying feasible solutions under strict resource constraints.

The STRCPN-based simulation approach guarantees the feasibility of the generated solutions; however, it is inherently highly stochastic. Repeating deterministic simulations (with fixed durations and resource profiles) may yield satisfactory solutions by chance, but for larger projects involving dozens or hundreds of activities, such blind sampling becomes infeasible.

The schedule can be represented as a vector:

$$h = (h_1, h_2, \ldots, h_n) \tag{2}$$

where: h – schedule vector, h_i denotes the start time of activity i.

The proposed solution for controlling the simulation trajectory and reducing its randomness is the introduction of a delay vector:

$$d = (d_1, d_2, \ldots, d_n) \tag{3}$$

where: d_i is the lower bound on the start time of activity i.

When the delay vector d is set to zero, the simulation proceeds according to standard Petri net rules, and the resulting schedule may be highly suboptimal. By introducing a positive delay $d_i > 0$, we eliminate all schedules that violate the condition:

$$h_i \geq d_i \tag{4}$$

Thus, the delay vector allows restricting the simulation to a controlled subset of feasible schedules. Increasing delay values narrows the search space. An extreme case occurs when the delay vector is identical to one of the previously generated schedules, effectively fixing the simulation to that specific solution.

In the Petri net simulation model, delays are introduced as dummy activities preceding the corresponding process activities. Each dummy activity:

- has no predecessors,
- starts at time 0,
- has a duration equal to the imposed delay,
- does not consume any resources.

We propose the DIM, which uses the concept of the delay vector to guide the optimization of project schedules. The method performs recursive simulations, progressively searching for improved schedules by exploring neighborhoods around the best solutions found in earlier iterations (see Figure 5).

The optimization process proceeds as follows:

A – For a randomly selected project variant, resource consumption is established, and activity durations are sampled based on the assigned probability distributions.

B – In the initial optimization step, a set of delay vectors with all-zero entries is provided as input to the simulation. The number of initial vectors is a configurable control parameter of the algorithm.

C – For each delay vector, S simulation runs are executed, and the best schedule (in terms of process completion time) is selected from the generated stochastic results.

D – From the pool of simulated results, the top N (number of input vectors) best-performing schedules are selected, again based on the minimum makespan criterion.

E – For each of the N selected schedules, a new delay vector is computed using the following formula:

$$\forall i = 1,2,\ldots,n: d_i^{incr} = d_i + v(h_i - d_i) \quad (5)$$

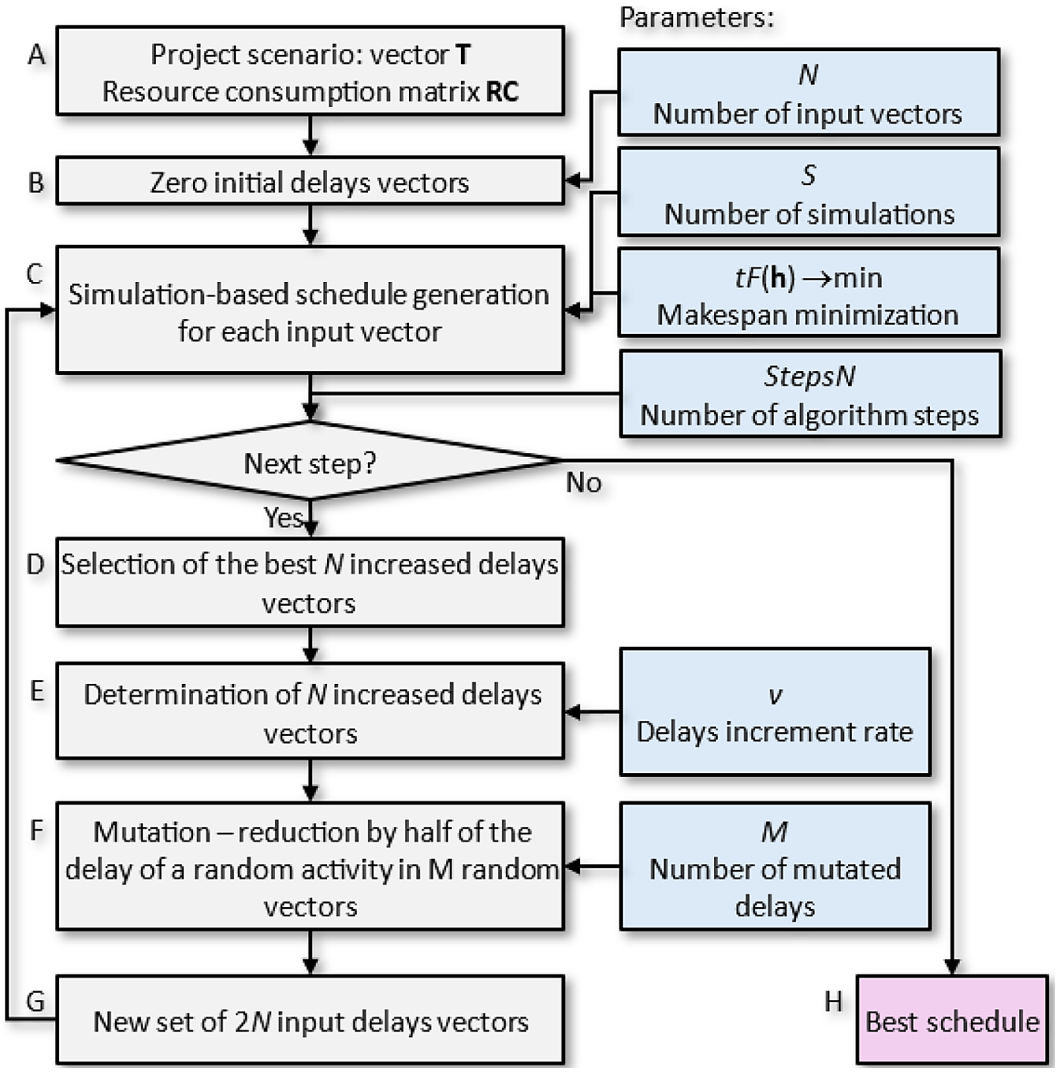


Figure 5. DIM algorithm

where: d_i is the current delay for activity i; h_i is the start time of activity i obtained from the simulation; d_i^{incr} is the updated delay for activity i; $v \in [0,1]$ is a delay increment velocity parameter.

All computed delays are rounded to the nearest integer.

F – To avoid premature convergence, a mutation-like operator is applied. A randomly chosen subset of $M \leq 2N$ (M – number of mutated delays) delay vectors undergoes delay reduction, where the delay for one randomly selected activity is halved. This mechanism, inspired by evolutionary algorithms, introduces variability and diversifies the search space.

G – The N best delay vectors and the incremented vectors are combined to form a new input population for the next simulation iteration.

H – After a predefined number of iterations, the algorithm terminates and returns the best schedule along with its project completion time.

An illustrative example of the DIM algorithm's operation and a performance analysis are provided in the next section. Before deploying the DIM algorithm in practice, a parameter calibration procedure must be carried out. These parameters must be tuned individually for each problem instance, and the proposed calibration method is also demonstrated using the example.

Once the optimal schedule has been generated, it is passed on to execution, during which continuous monitoring of activity start times is required. If delays occur relative to the planned start times, an analysis is conducted to evaluate their impact on the remaining project plan. If necessary, subsequent activity start times are postponed accordingly.

Preliminary analysis of algorithm performance

In research on the RCPSP, a common practice is to validate proposed methods using the PSPLIB test data [35], which for decades has served as a recognized benchmark for optimization algorithms. The instances enable comparison of the quality and efficiency of different methods on a uniform dataset. They assume deterministic activity durations, which prevent direct comparison with stochastic models. Therefore, as an initial step, we test the DIM method on deterministic PSPLIB instances to demonstrate its basic effectiveness and to enable reference to results widely reported in the literature.

For this purpose, we employed the RG30 dataset developed by Vanhoucke et al. (2008) [37] using the RanGen generator, which is described in detail in the work of Demeulemeester et al. [38]. From this dataset, one instance was selected from each set, resulting in a total of five instances. The makespan values obtained using the DIM method were compared with the results of the genetic algorithm (GA), scatter search (SS), and electro magnetic (EM) algorithm [38]. The differences in reference values between the compared methods were calculated, and the results are presented in Table 2.

The values obtained by the selected algorithms are presented as means accompanied by standard deviations, calculated over 1.000 schedules in 40 trials. In a separate column, the lowest value achieved by the algorithms was also reported, based on 500,000 schedules generated in 100 trials. To enable a fair comparison with the reference metaheuristics, it was necessary to account for the computational budget, understood here as the total number of simulations. For the DIM algorithm, this number was determined according to the following formula:

$$TS = N + (2 \cdot N \cdot S \cdot StepsN) \tag{6}$$

The algorithm parameters were selected such that the total number of evaluated schedules matched the computational budget reported for the other algorithms.

The results obtained with the DIM algorithm are competitive with those of the compared methods. The makespan values achieved by DIM were lower than the averages obtained by GA by approximately 0.6%, by SS by about 0.9%, and by EM by around 1.15%. When considering the best results, only for instance Set2/Pat1 did DIM produce an outcome identical to the other methods. Overall, the solutions generated in a single run of DIM were qualitatively very close to those obtained through multiple repetitions of GA, SS, and EM.

APPLICATION OF THE DIM ALGORITHM IN SHIPYARD PRODUCTION SCHEDULING

Example problem definition

Shipyard production encompasses the processes of design, part fabrication, hull assembly,

Table 2. Results of DIM versus GA, SS, and EM on RG30 instances

Instance from RG30	DIM	avgGA (stdGA)	minGA	Diff [%]	avgSS (stdSS)	minSS	Diff [%]	avgEM (stdEM)	minEM	Diff [%]
Set1/Pat1	64	64.6 (0.7)	62	-0.93	65.1 (0.7)	62	-1.69	65.3 (0.8)	62	-1.99
Set2/Pat1	58	58.7 (0.8)	58	-1.15	59.1 (0.6)	58	-1.82	59.2 (0.8)	58	-2.03
Set3/Pat1	76	75.9 (0.4)	75	0.20	76.0 (0.2)	75	0.07	76.0 (0.4)	75	0.07
Set4/Pat1	48	47.9 (0.5)	46	0.26	48.0 (0.5)	46	-0.05	48.3 (0.5)	46	-0.52
Set5/Pat1	78	79.0 (0.2)	78	-1.23	78.7 (0.5)	78	-0.92	79.0 (0.5)	78	-1.30

outfitting, launching, and sea trials. The developed algorithm was tested on an example project representing the fabrication, prefabrication, and final assembly phases of a welded structure, such as the hull of a small vessel.

The project sequence is illustrated in Figure 6. It begins with the cutting of steel materials, including plates and hot-rolled profiles. During part fabrication, the edges are beveled for welding, and the geometry of the parts is shaped.

The final product, referred to as the 'final assembly', is formed by joining two subassemblies (no. 8 and 9) and a part no. 11. The prefabrication stage is divided into two phases:

- the panel assembly phase (activities 28–33), consisting of welding together plates and profiles,
- and the block assembly phase (activities 34–36), in which the preassembled panels are integrated into larger structural blocks.

Prefabrication processes primarily involve positioning structural components and executing weld joints.

The process model includes CC activities following each prefabrication step. Each CC activity is followed by a transport operation. Panels are moved using a self-propelled platform (activities 47–52), while blocks are transported via a rail-mounted crane (activities 53–55).

The adopted process sequence is inspired by real-world shipyard fabrication and assembly workflows but includes simplifications to ensure clarity and conciseness. These simplifications were necessary due to space limitations of the article; however, the model captures the core logic of actual shipyard operations.

The modeled process includes 55 activities, which enables a meaningful evaluation of the computational performance of the proposed optimization algorithm. For each activity, the expected duration values and variances were assumed (Table B1, Appendix B). These are dimensionless values, selected to appropriately illustrate the proposed modeling and optimization methods. Particular attention was paid to maintaining proportions between activity durations across different stages to reflect the general nature of shipyard processes.

The project involves the use of the following types of resources:

- renewable resources, including workers, workstations, and transportation equipment;
- non-renewable resources, such as steel materials, welding supplies, and outfitting components for structural sections.

A schematic representation of the project flow, along with the numbered resource types, is shown in Figure 7. Arcs in the diagram indicate resource flows, and the resource demand is annotated accordingly. Activities within the same project phase are characterized by identical resource requirements. However, certain operations in the panel assembly stage require increased quantities of welding materials.

CC activities consume resources depending on the selected execution mode. In the case of control-only mode, the activity requires the use of a dimensional control team. When corrective actions are needed, additional resources are engaged, including welders.

Algorithm calibration

The calibration of the DIM algorithm's control parameters was performed for the previously introduced example project. The test scenario was developed based on a randomly generated sample consisting of 1000 project realizations.

The test durations were computed using Equation 1 with the risk coefficient $k = 3$, corresponding to a risk-averse scheduling strategy. The resulting values are presented in Appendix B, Table B3.

The test scenario was used to calibrate the algorithm's parameters in order to assess their influence on solution quality and computational

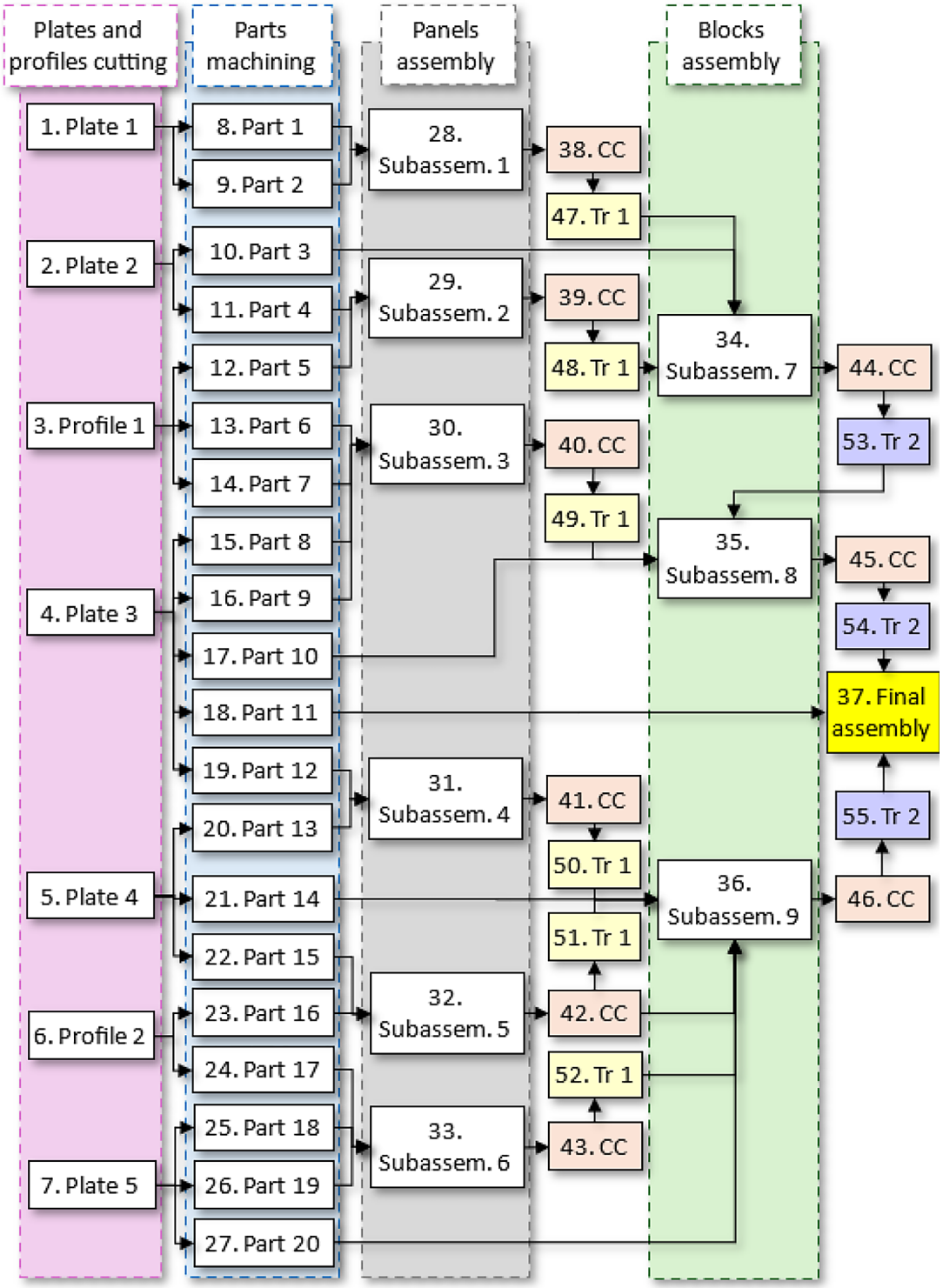


Figure 6. Activities of the example project; CC – control and/or corrections, Tr 1 – transport by a self-propelled platform, Tr2 – transport by a crane

performance. The parameters shown in Figure 5 were varied independently, while the remaining parameters were kept constant. For each configuration, the algorithm was executed five times. The project completion time *tF* (makespan) and total computation time were used as evaluation criteria.

Table 3 presents the investigated parameter ranges. These ranges were selected based on preliminary observations of the algorithm's behavior. The adopted constraint was that the runtime of a single optimization should not exceed 1000 seconds. This runtime is strongly dependent on the total number of simulations performed during

the computations. The selected results shown in Table 3 correspond to the lowest values of *tF* obtained during the calibration process.

The conducted experiment demonstrated that the total number of simulations directly affects the algorithm's execution time, Figure 8 illustrates this relationship. The course of each simulation is partly unpredictable due to the random activation of transitions competing for tokens in the Petri net. Consequently, the optimization runtime is partly stochastic, and this effect becomes more pronounced as the number of simulations increases. Nevertheless, the overall trend remains clear.

Increasing the number of input vectors N results in a faster growth of the total number of simulations TS than increasing the number of simulations S per vector. Moreover, a larger value of N does not provide a clear improvement in solution quality. The number of simulations S performed for each delay vector influences solution quality but results in higher computational cost. The large variability of S among the best-performing solutions indicates that the key parameters are the number of optimization steps and the delay increment velocity.

When increasing the number of optimization steps $StepsN$, the delay increment velocity v was simultaneously reduced. $StepsN = 8$ and $v = 0.1$ proved to be particularly promising, enabling relatively fast convergence to high-quality solutions. The computation time of only a few minutes for such a large-scale case is considered very low.

RESULTS

Optimal schedule for selected strategies

For the described shipyard project, the scheduling procedure was applied. Two planning strategies were tested, and an optimal schedule was generated for each of them. The DIM algorithm was executed with a fixed configuration established during the calibration phase, ensuring a balance between solution quality and computational effort. The two schedules were used to determine the planned project complectiont imes, as shown in Figure 9, where activities are shown in ascending order from bottom to top.

The distribution of activity intensity differs between the two schedules. In the case of the risk-averse strategy (Figure 9a), the highest concentration of activity is observed between time units 10 and 110. For the risk-seeking strategy (Figure 9b), the most intense workload occurs between time units 10 and approximately 80. This approach results in a condensed execution of activities at the beginning of the schedule, with many tasks carried out in parallel over a shorter period. It enables the project to be completed in 210 time units, but with a smaller temporal buffer. In contrast, the risk-averse strategy yields a completion time of 243, with longer activity durations leading to a more extended period of intensive work.

The resource utilization profile over time follows a similar pattern. During the initial and middle phases of the project, the usage of renewable resources approaches their availability limits. This is particularly evident for welders, whose limited availability constrained the number of concurrent operations. In the later stages of project execution, the utilization of renewable resources decreases, while the consumption of non-renewable resources increases.

Comparative analysis of planning strategies

The obtained schedules were evaluated in terms of their robustness to stochastic activity

Table 3. Results of DIM parameters tuning

Parameters					Results		
N	S	v	StepsN	M	tF	Elapsed time [sec.]	Total no. simulation
[5,20]	[5,50]	[0.1;0.9]	[2,10]	[1,3]			
10	10	0.1	8	2	245	110,257	1610
10	40	0.1	8	2	245	400,383	6410
10	10	0.3	8	2	245	137,778	1610
10	20	0.3	8	2	245	259,469	3210
10	30	0.3	10	2	245	544,556	6010
10	40	0.3	10	2	245	702,661	8010
10	50	0.3	8	2	245	675,185	8010
10	40	0.5	8	2	245	596,721	6410

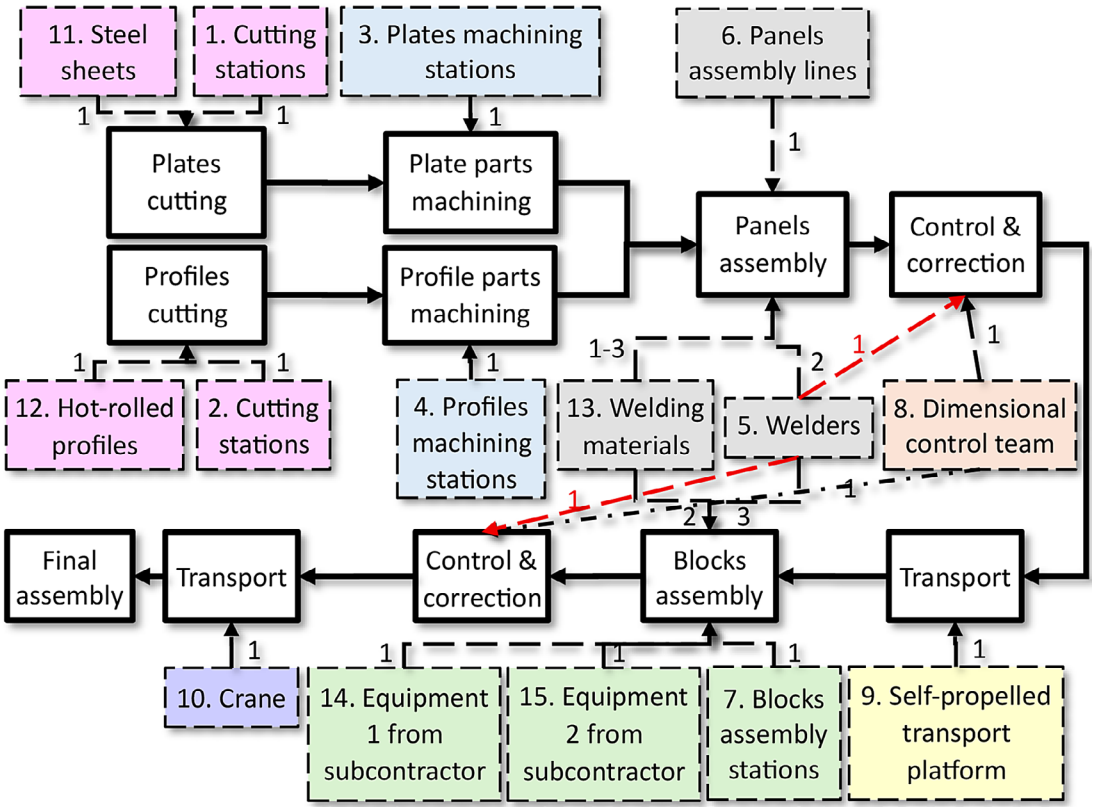


Figure 7. Resources required at different stages of the project

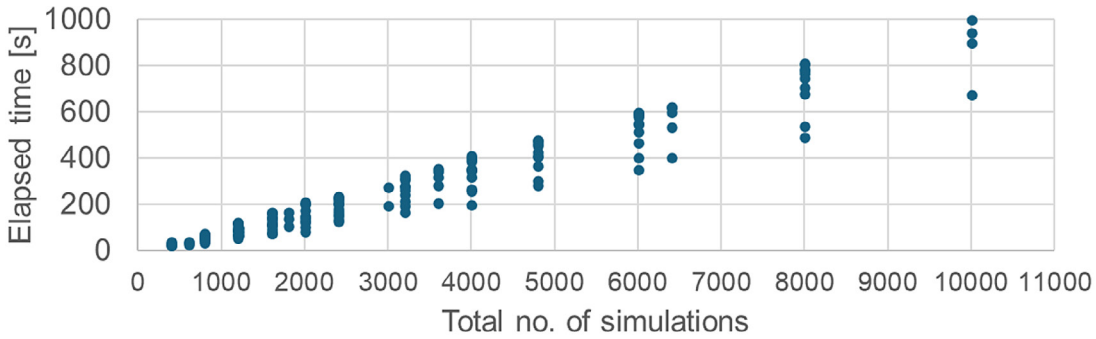
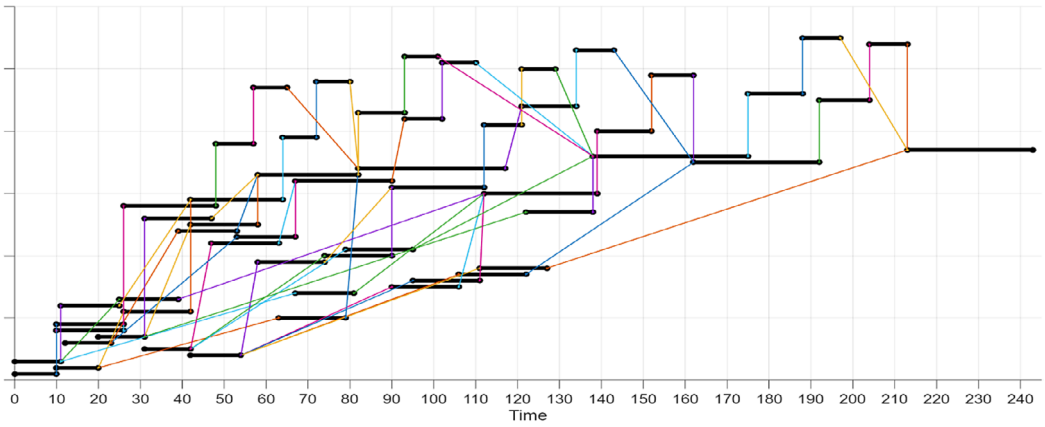


Figure 8. The dependence of computation time on the total number of simulations

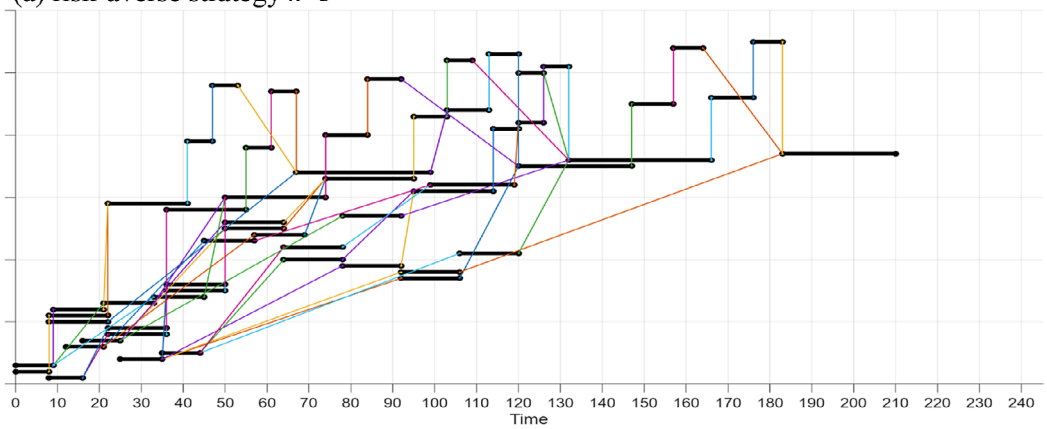
durations and potential delays. For this purpose, 1.000 random duration scenarios were generated for each of the two schedules, and a disturbance-resistance analysis was conducted to determine the resulting project completion times *tF*. For each case, the probability of meeting the optimal *tF* was calculated and is presented in Figure 10.

As expected, the schedule corresponding to the risk-averse strategy demonstrates greater resilience to random variations in activity durations and if activity durations are shortened, the project can be completed within 237–239 time units under this scenario. In 1000 random simulations, the threshold $tF = 243$ was exceeded 2 times, yielding an estimated exceedance probability of 0.002 (0.2%). Based on the normal approximation, the 95% confidence interval was calculated as [-0.0007, 0.0047], which was subsequently truncated to the admissible probability range [0, 0.0047]. Thus, with 95% confidence, the true probability of exceeding the threshold is below 0.5%.

In contrast, the risk-seeking strategy yields a shorter planned project duration but shows less robustness to variability in activity durations. According to the analysis, the most probable project completion times for this schedule fall between 207 and 209 time units. The threshold



(a) risk-averse strategy k=3



(b) risk-seeking strategy k=1

Figure 9. Gantt charts of the optimal schedules obtained by the DIM

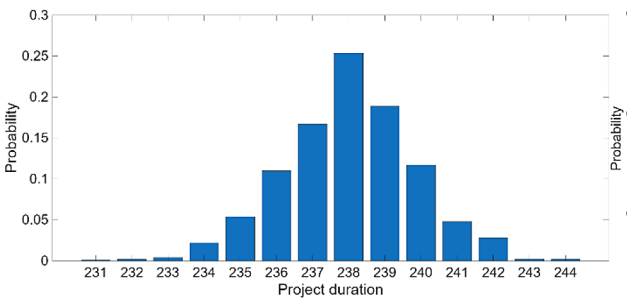
was exceeded 94 times, resulting in an estimated exceedance probability of 0.094 (9.4%). The standard error was 0.0092. The 95% confidence interval was computed as [0.0759, 0.1121]. Thus, with 95% confidence, the true probability of exceeding the threshold lies between approximately 7.6% and 11.2%.

The optimal schedules obtained for both strategies can be considered stable solutions, as confirmed by the low risk of deadline overrun and the narrow range of most likely completion times.

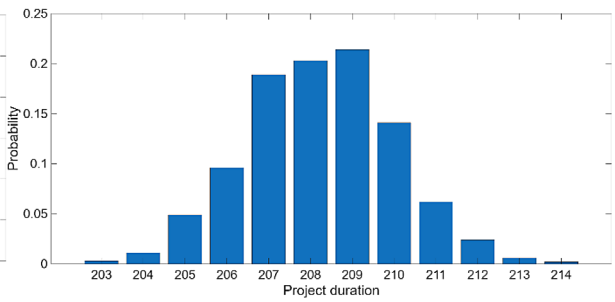
DISCUSSION

The calculated number of simulations performed within the DIM algorithm enables an evaluation of its computational efficiency. This evaluation was carried out by comparing the minimal project completion time obtained using DIM with the results of random simulations of the STRCPN model.

The number of simulations was aligned with the best-performing configuration used in the



(a) pessimistic strategy



(b) optimistic strategy

Figure 10. Probability distribution

optimization of the shipyard case study. In total, 7020 STRCPN simulations were executed, taking 65.8 seconds to complete. The best result achieved was $tF = 249$, which occurred only once; all other results exceeded 251 time units (Figure 11). The average project completion time across these runs was 297, with a standard deviation of 18. The box plot illustrates that the distribution of project completion times for STRCPN is wide, with a median of 297 and a spread ranging from 249 to 341. In contrast, DIM results are tightly clustered between 243 and 250, with a median of 246.

These results demonstrate that the DIM algorithm outperforms all solutions obtained through random STRCPN simulation. This confirms that the developed algorithm explores the solution space in a purposeful and guided manner, rather than relying on chance.

The proposed decision-making process enables the definition of acceptable risk levels in project schedules through the use of a risk coefficient k. By selecting different values of this parameter, alternative planning strategies can be generated for a given project. Subsequent evaluation of these strategies, based on the probability of completing the project within the planned deadline, allows for a comparative analysis of different approaches to schedule risk management. This supports the selection of a strategy tailored to the specific characteristics of the project, grounded in statistical analysis of simulation outcomes. The procedure facilitates the exploration of a broad spectrum of realistic planning variants, providing practical support in operational scheduling.

For a selected project scenario, the optimal schedule is generated using the proposed algorithm. The DIM effectively addresses the stochastic nature of STRCPN simulations by introducing a delay vector, which reduces the randomness of outcomes. The developed metaheuristic iteratively narrows the space of feasible schedules, thereby improving solution consistency and quality across simulation runs.

The application of DIM to STRCPN simulations for projects involving dozens or more activities introduces computational overhead. Each iteration involves multiple simulations of the Petri net for each delay vector, which can lead to significant memory and processor usage in large problem instances. Tests conducted on computers with varying configurations have shown that, for systems with 32 or 64 GB of RAM and an Intel i7-class processor, the runtime for the example project remained within a few minutes. For smaller instances, the runtime was often under one minute. In contrast, computers with 16 GB of RAM and an AMD Ryzen 5 processor experienced an average 50% increase in computation time. From a practical standpoint, the use of DIM is recommended on machines equipped with at least 32 GB of RAM, especially for projects with many activities and multiple resource constraints.

The DIM method was implemented in MATLAB, which requires each project instance to be formalized in matrix form. For this reason, the method is particularly effective in contexts where a structured digital process representation is available. In cases where input data is not readily available – for example, due to the absence of an ERP-class system – the modeling effort may become time-consuming and may limit the method's applicability in low-digitalization environments.

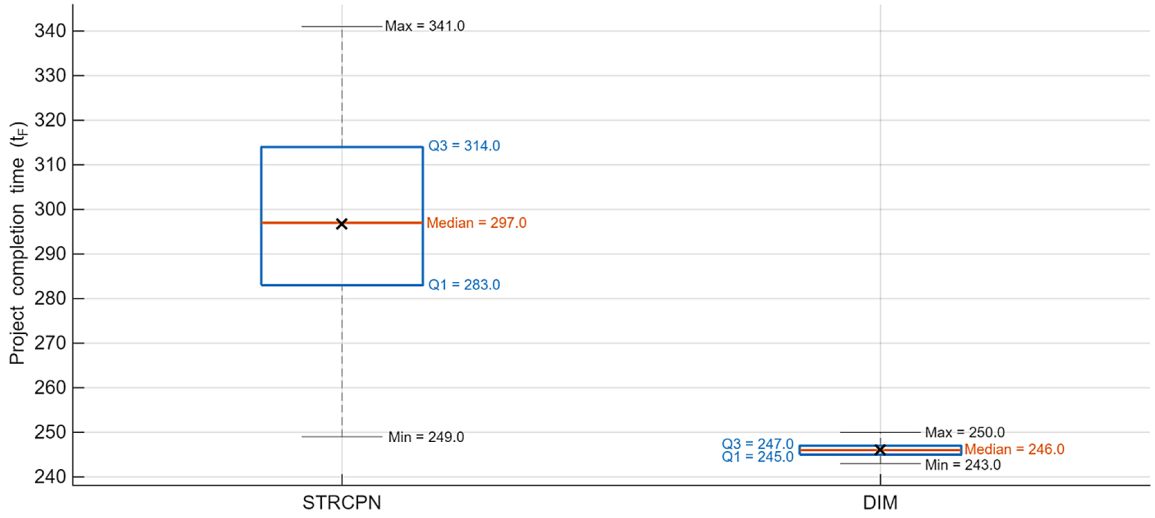


Figure 11. Box plot of comparison STRCPN vs. DIM algorithm

The output of the algorithm includes not only the optimal schedule (as a vector of activity start times), but also an automatically generated Gantt chart. This visualization supports the analysis of the computed schedule and enhances the clarity of result presentation. In practice, the DIM method enables the automation of schedule generation, significantly reducing planning time and increasing the efficiency of the scheduling process.

CONCLUSIONS

The foundation of the proposed project scheduling method under resource constraints is the developed STRCPN model. This simulation framework accurately captures activity durations, project sequencing, and the dynamics of resource supply and consumption. It has been demonstrated that the simulation model correctly generates random yet feasible schedules.

The innovative contribution of this study is the introduction of the DIM, which enables directed simulation of the STRCPN model toward solutions with desired characteristics. DIM is an evolutionary approach that generates new solutions based on the best-performing ones from the previous iteration and includes a mutation operator. The method exhibits convergence, as solutions improve more rapidly over time compared to purely random search.

As the authors aimed to develop a decision support tool under uncertainty, the paper also proposes a way to apply the DIM algorithm depending on the selected risk management strategy. This strategy is expressed in the model through a single control parameter.

The simplification of the supply model was necessary to ensure greater clarity in the analysis of the proposed optimization method. Scheduling based on Petri nets enables the search for solutions within a domain defined by the allocated level of renewable resources and the assumed deterministic plan of non-renewable resource deliveries. Such a supply system, however, deviates from real-world conditions, where:

- resource levels may vary randomly due to unforeseen events,
- deliveries may be delayed,
- supplied prefabricated components and materials may turn out to be defective.

Advancing the model toward incorporating stochastic delivery times and fluctuating resource levels will require the collection of substantial amounts of additional statistical data, close collaboration with experienced supply chain personnel, and coordination of information flows with suppliers. This extension will inevitably increase the uncertainty of the planning decision environment. These aspects call for further investigation.

The presented scheduling method is tailored to the shipbuilding industry. It accounts for the stochastic necessity of corrective actions, and it allows modeling of various resource provisioning schemes, including single deliveries of large, prefabricated hull blocks, main engines, and other specialized components characteristic of shipyard logistics.

The developed approach can be applied to planning the hull construction of a small vessel (such as a yacht, fishing cutter, or barge) or a single hull block of a large ship. In the case of a large shipyard, it may thus serve as a useful tool for departments responsible for detailed scheduling of prefabrication for each block independently. The synchronization of work between blocks, the scheduling of hull assembly station occupancy (dock or slipway), and the use of large gantry cranes constitute a separate optimization problem.

Based on the developed detailed schedules, it is possible to aggregate information on execution time and resource consumption for each block. This enables treating even a large-scale prefabrication project as a single activity at a higher level of planning. In this way, the shipyard's overall production schedule can be optimized on a broader scale. From a shipyard production perspective, a valuable extension could be the development of a method for synchronizing the planning of multiple concurrent projects at various stages of progress. This would enable the implementation of a global scheduling and resource management system for shipyards building multiple vessels simultaneously across different docks or slipways. However, such a recursive, multi-level planning method requires further investigation and represents a research challenge for the future.

The present study demonstrates the methodological foundations and computational feasibility of the proposed approach; however, several limitations remain. First, industrial validation has not yet been conducted. to disruptions and the capacity for real-time schedule adjustment.

Acknowledgements

All research used for this article has been performed using the infrastructure of The Centre for Operation of Floating Objects, part of Maritime University of Szczecin.

REFERENCES

1. Zhang X., Chen D. Shipbuilding 4.0: A systematic literature review. Appl. Sci. 2024; 14(14): 6363. https://doi.org/10.3390/app14146363
2. Song T., Zhou J. Research on Lean Shipbuilding and Its Manufacturing Execution System. J. Ship Prod. Des. 2022; 38(3): 172–181. https://doi.org/10.5957/jspd.01220001
3. Okamoto A. Studies concerning design and production system for shipbuilding in the past (2nd report). 2019.
4. Błażewicz J., Lenstra J.K., Kan A.H.G.R. Scheduling subject to resource constraints: classification and complexity. 1983.
5. Khajesaeedi S., Sadjadi S.J., Barzinpour F., Moghaddam R.T. Resource-constrained project scheduling problem: Review of recent developments. J. Project Manag. 2025; 10(1): 1–26. https://doi.org/10.5267/j.jpm.2024.12.002
6. Gómez Sánchez M., Lalla-Ruiz E., Fernández Gil A., Castro C., Voß S. Resource-constrained multi-project scheduling problem: A survey. Eur. J. Oper. Res. 2023; 309(3): 958–976. https://doi.org/10.1016/j.ejor.2022.09.033
7. Mao H., Yuan J. The performance of priority rules for the decentralized resource-constrained multi-project scheduling. Knowl.-Based Syst. 2024; 304: 112530. https://doi.org/10.1016/j.knosys.2024.112530
8. Wang M., Liu G., Lin X. Dynamic optimization of the multi-skilled resource-constrained project scheduling problem with uncertainty in resource availability. Mathematics 2022; 10(17): 3070. https://doi.org/10.3390/math10173070
9. Wang P., Lu S., Cheng H., Liu L., Pei F. Preemptive multi-skill resource-constrained project scheduling of marine power equipment maintenance tasks. J. Intell. Fuzzy Syst. 2023; 44(3): 5275–5294. https://doi.org/10.3233/JIFS-221994
10. Shahabi-Shahmiri R., Tavakkoli-Moghaddam R., Dolgui A., et al. Preemptive and non-preemptive multi-skill multi-mode resource-constrained project scheduling problems considering sustainability and energy consumption. J. Environ. Manag. 2024; 367: 121986. https://doi.org/10.1016/j.jenvman.2024.121986
11. Zhang H., Demeulemeester E., Li L., Bai S. Surrogate-assisted cooperative learning genetic programming for the resource-constrained project scheduling problem. Comput. Oper. Res. 2025; 173: 106816. https://doi.org/10.1016/j.cor.2024.106816
12. Zhou T., Long Q., Law K.M.Y., Wu C. Multi-objective stochastic project scheduling with alternative execution methods. Expert Syst. Appl. 2022; 203: 117029. https://doi.org/10.1016/j.eswa.2022.117029
13. Hatami-Moghaddam L., Khalilzadeh M., Shahsavari-Pour N., Sajadi S.M. Robust multi-skill, multi-mode resource-constrained project scheduling with partial preemption. Mathematics 2024; 12(19): 3129. https://doi.org/10.3390/math12193129
14. Ramos A.S., Miranda-Gonzalez P.A., Nucamendi-Guillén S., Olivares-Benitez E. A formulation for the stochastic multi-mode resource-constrained project scheduling problem. Mathematics 2023; 11(2): 337. https://doi.org/10.3390/math11020337
15. Zheng Z., Shumin L., Ze G., Yueni Z. Resource-constraint multi-project scheduling with priorities. 2012.
16. Zhou Y., Miao J., Yan B., Zhang Z. Stochastic resource-constrained project scheduling problem with time varying weather. Comput. Ind. Eng. 2021; 157: 107322. https://doi.org/10.1016/j.cie.2021.107322
17. Tao S., Wu C., Sheng Z., Wang X. Stochastic project scheduling with hierarchical alternatives. Appl. Math. Model. 2018; 58: 181–202. https://doi.org/10.1016/j.apm.2017.09.015
18. Tao S., Dong Z.S. Scheduling resource-constrained project problem with alternative activity chains. Comput. Ind. Eng. 2017; 114: 288–296. https://doi.org/10.1016/j.cie.2017.10.027
19. van der Beek T., van Essen J.T., Pruyn J., Aardal K. Exact solution methods for the resource-constrained project scheduling problem. Eur. J. Oper. Res. 2025; 322(1): 56–69. https://doi.org/10.1016/j.ejor.2024.10.029
20. Van Peteghem V., Vanhoucke M. Experimental investigation of metaheuristics for the multi-mode RCPSP. Eur. J. Oper. Res. 2014; 235(1): 62–72. https://doi.org/10.1016/j.ejor.2013.10.012
21. Liu W., Zhang H., Chen Y., Qu C., Zhang J. Simulation-based hybrid genetic algorithms for stochastic multi-mode RCPSP. Appl. Soft Comput. 2024; 161: 111716. https://doi.org/10.1016/j.asoc.2024.111716
22. Satic U., Jacko P., Kirkbride C. Simulation-based approximate dynamic programming for stochastic multi-project scheduling. Eur. J. Oper. Res. 2024; 315(2): 454–469. https://doi.org/10.1016/j.ejor.2023.10.046
23. Medina-Garcia S., Medina-Marin J., Montaño-Arango O., Gonzalez-Hernandez M., Hernandez-Gress E.S. A Petri Net approach for business process modeling and simulation. Appl. Sci. 2023; 13(20): 11192. https://doi.org/10.3390/app132011192
24. Prashant Reddy J., Kumanan S., Krishnaiah Chetty O.V. Application of Petri nets and a genetic algorithm to multi-mode RCPSP. Int. J. Adv. Manuf.

Technol. 2001; 17(4): 305–314. https://doi.org/10.1007/s001700170184

25. Niño K.Y., Mejia G., Sanchez M., Figueroa P., Mejía G., Sánchez M., et al. A Petri Net based algorithm for RCPSP: A real life application in animation and videogame industry. 2013. Available at: https://www.researchgate.net/publication/259602597
26. Hu W., Wang H. A novel Petri-Net based resource-constrained multi-project scheduling method. In: Lecture Notes in Computer Science. 2014; 8630.
27. Storch R.L. Ship production. 2nd ed. Centreville: Cornell Maritime Press; 1995.
28. Okamoto A., Hirakata M. Studies concerning design and production system for shipbuilding (3rd report). 2020.
29. Makoto A., Nagaoka T. Optimization of hull block construction process by GA. 2001.
30. Skibińska K. Optimization system for workforce allocation in ship hull section assembly. Zesz. Nauk. Politech. Morsk. Szczecin 2025; 81(153): 196–211.
31. Li J., Lin P., Wu X., Song D., Yang B., Zhou L. Scheduling optimization of ship plane block flow line with dual resource constraints. Sci. Rep. 2024; 14(1). https://doi.org/10.1038/s41598-024-80785-5
32. Yu-guang Z., Kai X., Yong Z. Modeling and analysis of panel hull block assembly using timed colored Petri nets. Mar. Struct. 2011; 24(4): 570–580. https://doi.org/10.1016/j.marstruc.2011.07.002
33. Jeong D., Kim D., Choi T., Seo Y. A process-based modeling method for ship block assembly planning. Processes 2020; 8(7): 880. https://doi.org/10.3390/pr8070880
34. Iwańkowicz R. Optimization of assembly plan for large offshore structures. Adv. Sci. Technol. Res. J. 2012; 6(16): 31–36. https://doi.org/10.5604/20804075.1025126
35. Kolisch R., Sprecher A. PSPLIB—A project scheduling problem library: OR Software – ORSEP Operations Research Software Exchange Program. Eur. J. Oper. Res. 1997; 96: 205–216. https://doi.org/10.1016/S0377-2217(96)00170-1
36. Vanhoucke M., Coelho J., Debels D., Maenhout B., Tavares L.V. An evaluation of the adequacy of project network generators with systematically sampled networks. Eur. J. Oper. Res. 2008; 187(2): 511–524. https://doi.org/10.1016/j.ejor.2007.03.032
37. Demeulemeester E., Vanhoucke M., Herroelen W. RanGen: A random network generator for activity-on-the-node networks. J. Sched. 2003; 6(1): 17–38. https://doi.org/10.1023/A:1022283403119
38. RCPLIB project database by the Operations Research & Scheduling research group available at https://www.projectmanagement.ugent.be/research/data (accessed on 09.2025)