Archives of Civil and Mechanical Engineering (2021) 21:67
https://doi.org/10.1007/s43452-021-00218-2

ORIGINAL ARTICLE

Profit optimization for multi-mode repetitive construction project with cash flows using metaheuristics

Bartłomiej Sroka[1] · Jerzy Rosłon[2] · Michał Podolski[3] · Wojciech Bożejko[4] · Anna Burduk[5] · Mieczysław Wodecki[6]

Received: 19 November 2020 / Revised: 25 February 2021 / Accepted: 19 March 2021 / Published online: 7 April 2021

Abstract
The article presents the profit optimization model for multi-unit construction projects. Such projects constitute a special case of repetitive projects and are common in residential, commercial, and industrial construction projects. Due to the specific character of construction works, schedules of such projects should take into account many different aspects, including durations and costs of construction works, the possibility of selecting alternative execution modes, and specific restrictions (e.g., deadlines for the completion of units imposed by the investor). To solve the NP-hard problem of choosing the order of units' construction and the best variants of works, the authors used metaheuristic algorithms (simulated annealing and genetic search). The objective function in the presented optimization model was the total profit of the contractor determined on the basis of the mathematical programming model. This model takes into account monthly cash flows subject to direct and indirect costs, penalties for missing deadlines, costs of work group discontinuities, and borrowing losses. The presented problem is very important for maintaining a good financial condition of the enterprise carrying out construction projects. In the article, an experimental analysis of the proposed method of solving the optimization task was carried out in a model that showed high efficiency in obtaining suboptimal solutions. In addition, the operation of the proposed model has been presented on a calculation example. The results obtained in it are fully satisfying.

Keywords Optimization · Scheduling · Genetic algorithm · Simulated annealing · Flow shop · Time–cost trade-off · Repetitive construction projects

✉ Anna Burduk
anna.burduk@pwr.edu.pl

[1] Faculty of Civil Engineering, Cracow University of Technology, Warszawska 24, 31-155 Kraków, Poland

[2] Faculty of Civil Engineering, Warsaw University of Technology, Al. Armii Ludowej 16, 00-637 Warsaw, Poland

[3] Faculty of Civil Engineering, Wrocław University of Science and Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

[4] Department of Control Systems and Mechatronics, Faculty of Electronics, Wroclaw University of Science and Technology, Janiszewskiego 11-17, 50-372 Wrocław, Poland

[5] Faculty of Mechanical Engineering, Wroclaw University of Science and Technology, Łukasiewicza 5, 50-371 Wroclaw, Poland

[6] Department of Telecommunications and Teleinformatics, Faculty of Electronics, Wroclaw University of Science and Technology, Janiszewskiego 11-17, 50-372 Wrocław, Poland

1 Introduction

Scheduling is crucial for construction project planning. Properly prepared schedules allow construction managers to prioritize and allocate time for different tasks; represent the dependencies between them; manage resources, both renewable (labor, machinery, materials) and non-renewable (funds). To create a schedule, one must determine the order of tasks and their start times. This must be performed in such a way that specific objectives of the project are achieved whilst at the same time, organizational conditions are fulfilled, and planning constraints are applied. Some of the construction project planning objectives include minimizing the duration of the project; minimizing or providing evenness of the resources' consumption; maximizing the construction project economic value for the contractor. Organizational conditions may include, for example, the need to ensure the continuity of work performed by various groups of workers, or the possibility of using different construction activities

performance modes. Basic planning constraints are linked to the availability of the time, and resources needed to implement them and technological dependencies between the activities.

To meet the needs of the construction market, as well as to ensure flexibility in decision-making, the authors decided to combine concepts of flow shop and multi-mode resource-constrained project scheduling problem. The article presents a new optimization model of a multi-unit construction project with specific assumptions. Its most important assumptions are the need to perform many works in the undertaking for many different building structures, the availability of many different contractors for works in the facilities. The new scheduling model for a multi-unit construction project is illustrated by a computational example. The new presented model takes into account discounted cash flows, the knowledge of which is necessary for construction companies to make strategic decisions, and to survive on the market. In the model presented in the article, the whole project timespan is divided into n-day intervals representing billing periods (usually a month). Direct and indirect costs are calculated proportionally in each period and are discounted every *n*-days. Income was modeled in a form of Progress Payments (PP) which is common in the construction industry. The client pays contractors based on a monthly invoice, and report on completed works. The payments correspond to the direct and indirect costs plus the contractor's profit (and are also discounted). However, taking into account specifics of the construction industry, the authors assumed that the payments are made with a 1-month lag. To reflect the market characteristics, contractual penalties for exceeding the deadline were introduced into the model. Penalties related to the discontinuity of work of the work teams were also applied. These penalties are not discounted and are accounted for with a lag equal to one billing period. Accumulated cash flows are calculated for each *n*-day period. If the accumulated cash flow in a given period is negative, then we charge an additional penalty that corresponds to the cost of the loan required for the financing of the contractor's operations. The objective function assumes the maximization of the total profit—cumulated cash flow on the last day of the project's implementation (one period after completion of construction works).

2 Literature review

Theoretical studies of project scheduling problems are currently focused on searching for optimal schedules considering existing constraints. These are usually NP-hard optimization problems, as presented in [2, 3]. Due to the variety of possible constraints and schedules' objective functions, these problems can be grouped into different categories—models of project scheduling problems (PSP). Deterministic approach for solving PSP optimization is the most common one, due to its practical aspect. The state-of-the-art review of PSP deterministic models was presented in, e.g., [4–6]. One of the most common PSP models investigated by researchers is the well-known Resource-Constrained Project Scheduling Problem (RCPSP), e.g., [7]. This phenomenon results from the wide possibilities of applying this problem in the practice of industrial production.

Nowadays, in practice, more and more contracts focus on delivering non-standard products, or ones that were individually agreed on with the client. That is why scheduling production processes are often in line with the principles of project management. The same principle is also characteristic of the construction industry. A generalization of RCPSP is the Multi-Mode RCPSP (MRCPSP or rarely MMRCPSP). In such problems, each activity can be executed in one of the several modes. Each mode has a specific duration and specific resource requirements [8]. Due to the introduction of modes, decision makers are able to study different projects' variants; for example, testing how allocating additional resources to different tasks will affect project duration. It is worth mentioning that with the introduction of the additional decision variables (activities' modes), the possible solutions' space (and, at the same time, the amount of time required to solve the problem) increases. In other words, the computational time required for solving a MRCPSP is longer than that of a similar RCPSP without multiple modes [9].

Generalized RCPS/MRCPS problem is obtained by replacing the makespan minimization with other agents—any regular measure of performance: different types of trade-offs, objective functions, constraints, and conditions, e.g., total cost [10], NPV [11], quality [12], deviations from average employment level [13], total project delay [14], monthly cash demand [15], cost minimization in regard to the baseplan [16], and schedule robustness [17]. Such problems can be referred to as the Generalized Resource-Constrained Project Scheduling Problem (GRCPSP) [9]. Such problems can be divided into different categories, for example, the dependencies between time and cost of a project are taken into account in Multi-Mode Resource Constrained, Discrete Time–Cost Trade-Off problems (MRC-DTCTP) [13] which in construction are often called simply Time–Cost Trade-Off (TCT) problems. Other variations of the problem include P-MRCPSP (P at the beginning stands for Pre-emptive) in which activities can be pre-empted at any point in time and restarted at no additional cost [8] or MRCPSP with Discounted Cash Flows (MRCPSPDCF) [11], which focuses on maximization of NPV.

The procedures used to solve MRCPSP (and scheduling problems in general) can be classified as: exact, heuristic, and metaheuristic [1, 3]. The exact procedures include, among others, linear programming (LP), dynamic

programming (DP), and Branch and Bound method (B&B). The heuristic methods include priority rule-based heuristics [9].

Practical scheduling problems in construction can be easily qualified as NP-hard (non-deterministic, polynomial-time hard) problems. The time needed for solving such problems grows exponentially with the increase of the problem's size [2, 3, 9]—therefore, mathematical and heuristic methods often do not enable finding solutions to complicated construction problems within an acceptable period of time. In the view of many authors, metaheuristic algorithms seem to be the most appropriate measures for scheduling and task sequencing [1, 8–11, 18].

The metaheuristic approach does not guarantee finding the optimal solution and the obtained results are often subject to their input parameters; however, they seem perfect for solving complicated, NP-hard class problems because they enable computing suboptimal (acceptable) solutions within an acceptable time frame.

A great variety of metaheuristic algorithms can be used for solving GRCPSPs, e.g., Genetic Algorithms (GA) [8, 10], Simulated Annealing (SA) [11], Tabu Search (TS) [11, 19], Particle Swarm Optimization (PSO) [20], Ant Colony Optimization (ACO) [21] or hybrid algorithms [22, 23]. However, only a handful of conducted research activities, concerning multi-mode problems, revolve around the construction industry. Some of these cases involve real-life case studies. Özdamar and Dündar [24] optimized NPV for apartment building construction, with different modes' duration (however, activities' cost parameters were fixed). Chen and Weng [25], and Ghoddousi et al. [13] used GA to optimize time–cost trade-off for a simplified warehouse construction project. Zhang and Xu [26] minimized makespan (at the same time maximizing quality) of the hydropower plant using PSO algorithm. The same case has been optimized in terms of time, cost and quality, by Xu and Feng, with the use of hybrid algorithm [27]. Kulejewski and Rosłon reinforced the tabu search algorithm with artificial neural networks (ANN) to minimize the maximum monthly cash demand of apartment building construction [15]. Hegazy with other contributors minimized the total cost of multi-site [28] and repetitive projects [29]. Zhang et al. minimized duration for an example bridge project, their method was developed to facilitate repetitive project scheduling [30].

Methods of repetitive projects scheduling play important role in the current research regarding the construction project scheduling. Such projects can be easily partitioned into different units which can take a form of work zones, whole storys, building structures, or sections of construction objects characterized by the length (for example pipelines). Examples of the projects realized in a flow organization system include multi-story buildings, housing estates or groups of buildings, pipelines, and roads. Each unit of the project requires specialized work crews to perform assumed tasks. These work crews are moving from one unit to another [31]. A special case of repetitive projects is multi-unit projects, which include the implementation of residential, service, industrial or engineering constructions. In general, a feature of this type of project is the ability to set any order for the implementation of the works. Assuming any size of works that make up the implementation of a specific object in a multi-unit project, we get the opportunity to create the optimal schedule for the project. For $n=3$ objects, optimization tasks in this problem are NP-hard. To describe and solve such problems, we can use the flow shop theory of scheduling [32]. Research on the problem of multi-unit projects scheduling is focusing mainly on the improvement of current optimization models and improvement of optimization methods [33–37]. In [33], the problem of scheduling with minimization of penalties for exceeding the project's objects completion deadlines is considered and solved by the metaheuristic scatter search algorithm. In [34], the multi-unit project scheduling model takes into account the possibility of overlapping works in the facilities and is solved using the tabu search algorithm. The optimization criterion is the minimization of the project duration. In the article [35], the NP-difficult problem of scheduling a construction project was considered with the criterion of the sum of penalties for exceeding the deadline for building construction. The parameters of this project were represented by fuzzy numbers or random variables with a normal distribution or the Erlang distribution. In [36], a scheduling model was presented with the possibility of performing one type of work by more than one working group and with sequence relationships given by any graph. The optimization task in this model was solved using the tabu search algorithm. In [37], the scheduling model for a multi-unit project assumes a linear relationship between the time and the cost of carrying out the activity. The criterion of minimizing the total value of the project cost determined on the basis of the mathematical programming model, taking into account direct and indirect costs, costs of missing deadlines and costs of workgroup discontinuities. The optimization task in this model was solved using the modified simulated annealing algorithm.

3 Model of multi-unit project

The paper deals with the issue of construction project's profit maximization. Using the terminology of the scheduling theory, this problem can be described as follows: a set of n indivisible tasks is given (in the case of this paper—construction objects) that must be performed with m machines (in this paper—teams of working groups). Each of the n construction objects requires m types of operations (in this paper—works) to be performed by m teams of working

groups. Out of m teams of workgroups, only one workgroup must be selected to perform a given operation (work) in a given construction object.

It is assumed that the order of execution of construction objects for each of the teams of working groups is the same, i.e., it is a permutational problem. We also assume that each working group from among working group teams has a strictly defined duration t and cost c of the work. Between works on a given construction object, there may be technological breaks between works or there may be a partial overlap of works. Planning a schedule for carrying out such an undertaking, it is crucial to find its optimal schedule that takes into account the criterion adopted by the schedule planner (decision maker).

Finding the schedule of the project makes it possible to determine the order of execution of the objects and the set of choices for the methods of execution of works (the selection of working groups from teams of working groups). The criterion adopted in the project is the maximization of the total profit achieved in the project, more precisely the cumulative cash flow on the last day of the project implementation (an example of the cumulative cash flow chart, prepared in accordance with the principles set out in [37], is shown in Fig. 1—it takes into account the costs incurred during project's implementation: direct costs of working groups, indirect costs, contractual penalties for exceeding the deadline for completing the construction, penalties related to breaks in the work of working groups, and loan costs).

The presented problem is a generalization of the classical permutational flow shop problem, which is shown schematically in Fig. 2. In the terminology of the scheduling theory, it is denoted as the $F||C_{\max}$ problem, according to Graham's

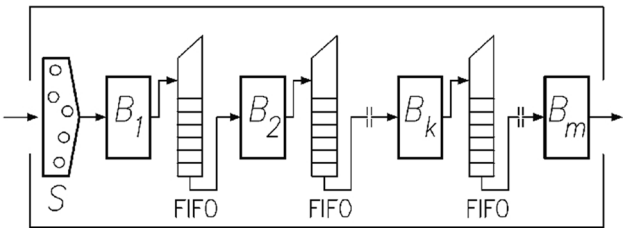

Fig. 2 Permutation flow shop system, where FIFO is the First In First Out

notation, presented in [39]. In Fig. 2, the variable S denotes the input sequence (permutation) of n tasks to the system of m machines. The cardinality of the set of possible solutions is $n!$.

3.1 Optimization model

The optimization model of the above described problem is as follows:

Parameters:

- The project consists of a set of building units

$$Z = \{Z_1, Z_2, Z_3, \ldots, Z_i, \ldots, Z_n\}.$$

- To carry out the project works the teams of working groups perform one job of one type which constitute the set

$$B = \{B_1, B_2, B_3, \ldots, B_j, \ldots, B_m\}.$$

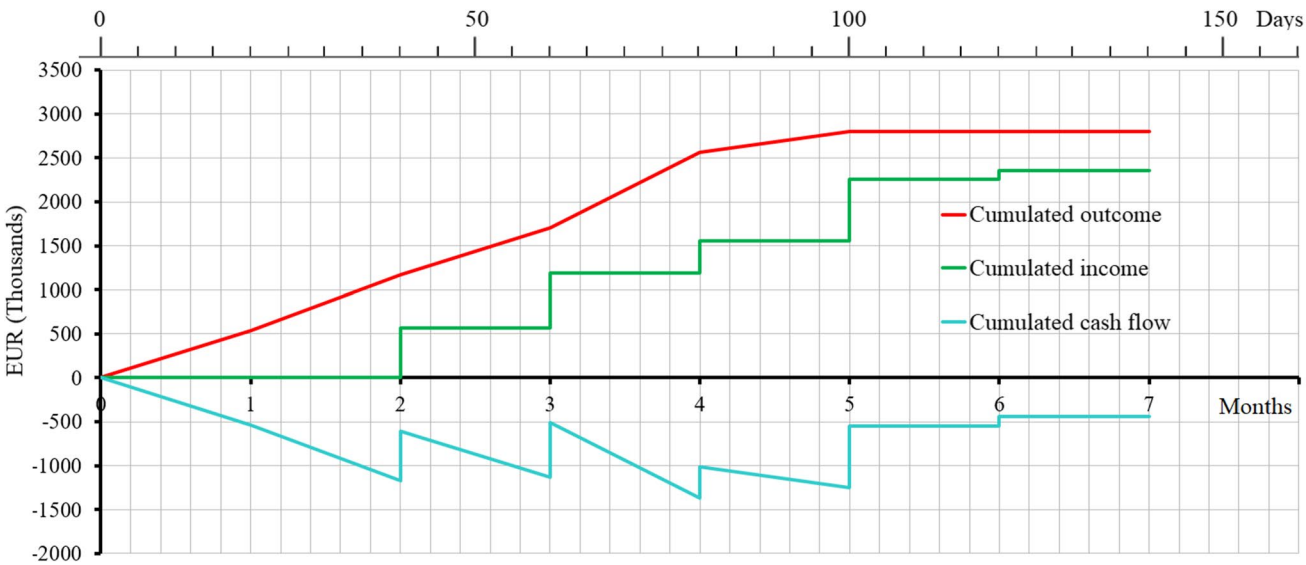

Fig. 1 Sample chart of cumulative cash flows in a construction project

- In each team of working groups $B_j \in B$, there are p working groups representing various subcontractors (modes with different capacities and configurations):

$$B_j = \{B_{j1}, \ldots, B_{jk}, \ldots, B_{jp}\},$$

where $k = 1 \ldots p$.
- Each object (unit) $Z_i \in Z$ requires implementation of m works which form the set

$$O_i = \{O_{i1}, O_{i2}, O_{i3}, \ldots, O_{ij}, \ldots, O_{im}\}.$$

- It is assumed that the work $O_{ij} \in O_i$ can be done by the working group $B_{jk} \subset B_j$. The duration of the work O_{ij} performed by the group is $t_{ijk} > 0$. The set of possible durations of works from the set O_i performed by the working group B_{jk} defines vector

$$t_i = [t_{i1}, t_{i2}, t_{i3}, \ldots, t_{ik}, \ldots, t_{im}],$$

where $t_{ij} = [t_{ij1}, \ldots, t_{ijk}, \ldots, t_{ijp}]$. Duration of works t_{ijk} are determined on the basis of workload and workgroup size (number of workers) B_{jk} performing the work O_{ij}.
- Similarly as the above, it is assumed that the work $O_{ij} \in O_i$ can be implemented by the working group $B_{jk} \subset B_j$. The cost of realization of the work O_{ij} by the working group B_{jk} defines a variable $c_{ijk} \geq 0$. The set of possible costs of works c_j from the set O_j is defined by the vector

$$c_j = [c_{j1}, c_{j2}, c_{j3}, \ldots, c_{jk}, \ldots, c_{jm}],$$

where $c_{jk} = [c_{ij1}, \ldots, c_{ijk}, \ldots, c_{ijp}]$.

The cost of the work c_{ijk} is determined by calculation of the cost of execution of the work O_{ij} by the working group B_{jk} located in the resources of the contractor. It may also be the cost offer of the execution of the work O_{ij} made by the subcontractor represented by the working group B_{jk}. It is assumed that the time of execution of works t_{ijk} are convex, decreasing cost functions c_{ijk}.
- There is the possibility of technological gaps between the works and the simultaneous operation of multiple working groups in the units assumed. Durations of intervals between a given work and the next work ($s_{ij} \geq 0$) or the length of the simultaneous duration of a given work and the next work ($s_{ij} < 0$) in the unit for a set of works O_i are given in vector $s_i = [s_{i1}, s_{i2}, s_{i3}, \ldots, s_{ik}, \ldots, s_{im}]$. These times should be understood as the minimum constraint and can take any value. In the further work, there will be called couplings between units.

Constraints:

- The order of execution of the works resulting from work technology is assumed such that

$$O_{i,j-1} \prec O_{i,j} \prec O_{i,j+1}.$$

- It is assumed that each working group from the team B_j can perform only one job at a time.
- It is assumed that the work $O_{ij} \in O_i$ is performed continuously by the working group $B_{jk} \subset B_j$ in time $t_{ijk} > 0$.

Decision variables are the order π of execution of units, which for each of the working group, is the same and takes the form of a permutation

$$\pi = (\pi(1), \pi(2), \pi(3), \ldots, \pi(j), \ldots, \pi(n))$$

and a set of numbers of ways of work execution (from $k = 1$ to $k = p$) in all units of the project is

$$R = (R_1, R_2, R_3, \ldots, R_i, \ldots, R_n), R \in \Re,$$

where $R_i = (R_{i1}, R_{i2}, R_{i3}, \ldots, R_{ij}, \ldots, R_{im})$, R_{ij} is the number of ways of realization (from $k = 1$ to $k = p$) of the work j in unit i, and R is the set of all possible ways to carry out the works in the project. The value of the number k of the way of realization R_{ij} enables allocation of the working group B_{jk} from the team B_j to the work j in the object i. The form of decision variable R uniquely identifies the allocation of working groups to realization of works in the units. Therefore, using the decision variable R, there are uniquely established not only the durations of individual works carried out in the units but also their cost. After the adoption of the decision variable, R the durations of works t_j from the set O_i is as follows:

$$t_i = \{t_{i1}, t_{i2}, t_{i3}, \ldots, t_{ij}, \ldots, t_{im}\},$$

where t_{ij} is the duration of the execution of the work j in the object (unit) i. Similarly, in consequence of the adoption of the decision variable R, the set of costs c_i of works from the set O_i is as follows:

$$c_i = \{c_{i1}, c_{i2}, c_{i3}, \ldots, c_{ij}, \ldots, c_{im}\},$$

where c_{ij} is the cost of implementing work j in the unit i.

The deadlines for the individual works for the decision variables π and R can be determined from the recursive formula:

$$F_{j,\pi(i)} = \max\{F_{\pi(i-1),j}, F_{\pi(i),j-1} + s_{\pi(i),j-1}\} + t_{\pi(i),j}, \quad (1)$$

where $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$, $\pi(0) = 0$, $F_{0,j} = 0$, $F_{i,0} = 0$.

The duration of the entire project $F_{n,m}$ (time execution of all works in the units) for $\pi^* \in \Pi$ and for the decision variable $R^* \in R$ is $F_{n,m}(\pi^*, R^*) = F_{\pi^*(n),m}$.

The deadlines for the performance of individual works and their cost can be found in time $O(nm)$. The number of possible solutions to the presented model is $n! \times p^{mn}$.

Objective function will be cumulated cash flow in the last billing period of the project which represents the total profit for the contractor. The cash flow will be maximized. Variables in the objective function are.

- $F_{i,j}$ is the finish time for works performed by a work crew j on unit i,
- $F_{n,m}$ is the finish time for construction works,
- h is the billing period, $h \in \{1, \ldots, H\}$, where H is the last billing period,
- TI is the time interval (in days), which represents the duration of each billing period,

For such adopted parameters, constraints and variables, there is a mathematical programming model formulated, in the following form:

$$\mathrm{CF}_{H+\max(d_{\mathrm{Inc}};d_{\mathrm{Pen}})} \rightarrow \max \tag{2}$$

s.t.:

$$H = \frac{F_{n,m}}{\mathrm{TI}} \tag{3}$$

$$\mathrm{CF}_h = \left(\mathrm{CF}_{h-1} - \mathrm{PC}_h + \mathrm{PV}_{h-d_{\mathrm{Inc}}} - \mathrm{PD}_{h-d_{\mathrm{Pen}}} - \mathrm{PW}_{h-d_{\mathrm{Pen}}}\right)\phi_h \tag{4}$$

$$\phi_h = \begin{cases} 1, & \text{if } \left(\mathrm{CF}_{h-1} - \mathrm{PC}_h + \mathrm{PV}_{h-d_{\mathrm{Inc}}} - \mathrm{PD}_{h-d_{\mathrm{Pen}}} - \mathrm{PW}_{h-d_{\mathrm{Pen}}}\right) \geq 0 \\ \left(1 + \mathrm{Pen}_{-\mathrm{CF}}\right), & \text{if } \left(\mathrm{CF}_{h-1} - \mathrm{PC}_h + \mathrm{PV}_{h-d_{\mathrm{Inc}}} - \mathrm{PD}_{h-d_{\mathrm{Pen}}} - \mathrm{PW}_{h-d_{\mathrm{Pen}}}\right) < 0 \end{cases} \tag{5}$$

- $t_{i,j}$ is the duration of activities performed by a work crew j to finish all required works on unit i,
- $t_{i,j,h}$ is the duration of works performed by a work crew j on unit i during billing period h,
- $t^W_{j,h}$ is the discontinuity of work duration for a work crew j during billing period h,
- t^D_i is the deadline for completion of works on unit i,
- $t^D_{i,h}$ is the delay for completion of works on unit i during billing period h,
- $c_{i,j}$ is the direct cost of activities performed by a work crew j to finish all required works on unit i,
- c^{Ind} is the indirect unit cost per day,
- α is the discount rate per time interval,
- Pro is the profit as a percentage,
- Pen^D_i is the unit cost penalty for a delay of works on unit i,
- Pen^W_j is the unit cost penalty for a discontinuity (downtime) of a work crew j,
- $\mathrm{Pen}_{\mathrm{CF}}$ is the penalty for negative cash flow as a percentage,
- ϕ_h is the binary variable for modeling negative cash flow penalties in billing period h,
- d_{Inc} is the delay (in time intervals) for income accounting,
- d_{Pen} is the delay (in time intervals) for penalties accounting,
- CF_h is the cumulated cash flow for billing period h,
- IC_h is the indirect costs in billing period h,
- DC_h is the direct costs in billing period h,
- PC_h is the production cost in billing period h,
- PV_h is the production value in billing period h,
- PD_h is the penalty for a delay of works in billing period h,
- PW_h is the penalty for a discontinuities (downtime) of a work crews in billing period h.

$$\mathrm{PC}_h = \frac{\mathrm{IC}_h + \mathrm{DC}_h}{(1+\alpha)^h} \tag{6}$$

$$\mathrm{IC}_h = \begin{cases} c^{\mathrm{Ind}}\mathrm{TI}, & \text{if } h < H \\ c^{\mathrm{Ind}}\left(F_{n,m} - \mathrm{TI}(h-1)\right), & \text{if } h = H \end{cases} \tag{7}$$

$$\mathrm{DC}_h = \sum_{i=1}^{n}\sum_{j=1}^{m}\frac{c_{i,j}}{t_{i,j,h}} \tag{8}$$

$$\mathrm{PV}_{h-d_{\mathrm{Inc}}} = \frac{\left(\mathrm{IC}_{h-d_{\mathrm{Inc}}} + \mathrm{DC}_{h-d_{\mathrm{Inc}}}\right)(1+\mathrm{Pro})}{(1+\alpha)^{h-d_{\mathrm{Inc}}}} \tag{9}$$

$$\mathrm{PD}_{h-d_{\mathrm{Pen}}} = \sum_{i=1}^{n}\mathrm{Pen}^D_i t^D_{i,h-d_{\mathrm{Pen}}} \tag{10}$$

$$\mathrm{PW}_{h-d_{\mathrm{Pen}}} = \sum_{j=1}^{m}\mathrm{Pen}^W_j t^W_{j,h-d_{\mathrm{Pen}}}. \tag{11}$$

The objective function (2) maximizes cumulated cash flow in the last billing period of the project which represents the total profit (TP) for the contractor. The number of billing periods (3) is calculated on the base of the time interval and finish time for construction works. Equation (4) presents the iterative method for cumulated cash flow calculation in the following billing periods.

Negative cash flow results in a necessity for finding additional funds for the conduction of the project's works. For contractors, this means taking interest-bearing loans or losing the option of investing their own funds (lost profits). Binary variable (5) is modeling this phenomenon by granting penalties in each billing period with negative cash flow.

Production cost (6) in each period is a discounted sum of indirect (7) and direct (8) costs in this period. Production value is invoiced after completion of a billing period and is received (income) by the contractor from the client with an agreed delay (d_{Inc}). Production value is subject to discount. Penalties for a delay of works (10) and discontinuities of work crews (11) are also accounted with a delay. First types of penalties are specified in the contract between the investor and the general contractor and usually amount to between 0.05 and 0.2% of the gross contract value for each day of delay. Downtime penalties are related to the contractor's need to keep the brigades ready for work at all times.

The described model can be represented in the form of a disjunctive graph $G(\pi)$, an example of which, with highlighted critical path is shown in Fig. 3. The form of the graph is dependent upon an established decisive variable π: $G(\pi)=(N, E(\pi))$,where N is a set of nodes, $E(\pi)$—set of arcs. It is assumed that $N=\{1,..,i,\ldots,n\}\ \{1,\ldots,j,\ldots,m\}$ is a set of nodes representing works in units of project. Weight of node (i, j) is equal to the time of work execution $t_{\pi(i),j}$. The set of $E(\pi)=E^F\ E^S(\pi)$ depends on the assumed decisive variable π. Horizontal arcs (sequential, representing processing order of units) from the set $E^S(\pi)$ are between nodes $\pi(i\text{-}1)$ and $\pi(i)$, where $i=1,\ldots,n$. Vertical arcs (technological) from the set E^F are between node standing for work j and j-1. The weight of the vertical arc from set E^F is the coupling between units $s_{\pi(i),j}$.

The above presented model is NP-hard optimization problem, because of assumptions from permutation flow shop problem (problem $F||C_{\max}$), which is strongly NP-hard. To solve the optimization task, there is individual algorithm proposed. This algorithm will use approximate metaheuristic simulated annealing algorithm or genetic search algorithm.

4 Optimization methods for the presented multi-unit model

The optimization model for the multi-unit project presented above is an NP-hard problem of discrete optimization. There are two different decision variables affecting the value of the objective function. This causes the need to construct an effective individual algorithm for solving discrete optimization tasks in the presented problem. The first decision variable is the order of execution of the project objects (units), which is represented by permutation of the execution of the objects π. This decision variable is discrete, and the issue of searching for the optimal order of execution of the units is NP-hard. Due to the small number of units (most often the maximum number of objects is a dozen or so) that make up the multi-unit project, it is proposed to use the simulated annealing algorithm to solve the sequencing optimization task.

The second decision variable is a set of numbers presenting modes of performing works by available contractors. This decision variable generates a much larger area of possible solutions depending on the number of objects, types of works and the number of modes of performing works for each element of a given unit. Therefore, in this article, it is proposed that this decision variable will be solved using a suitably adapted simulated annealing algorithm or genetic search algorithm.

The method of solving the discrete optimization task for the model of a multi-unit project presented in chapter 3, taking into account both different decision variables, is presented in the form of block diagrams in Figs. 4 and 5. Its steps are consistent with SA algorithm presented in chapter 4.1. When calculating the adopted objective function for a given order π, the problem

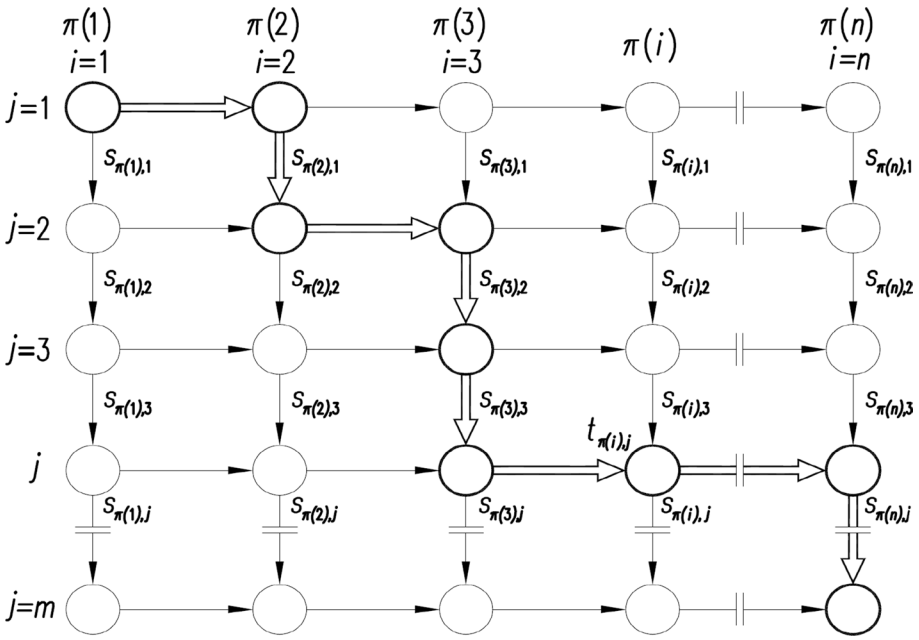

Fig. 3 The graph for the described model of the project with the marked critical path

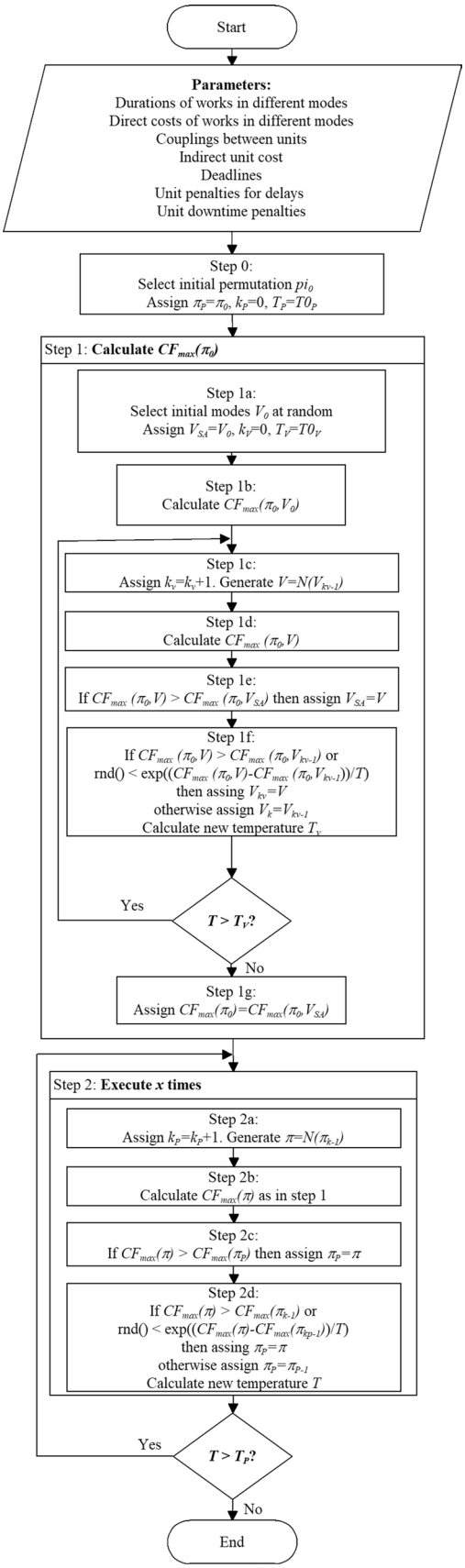

Fig. 4 SA + SA algorithm block diagram

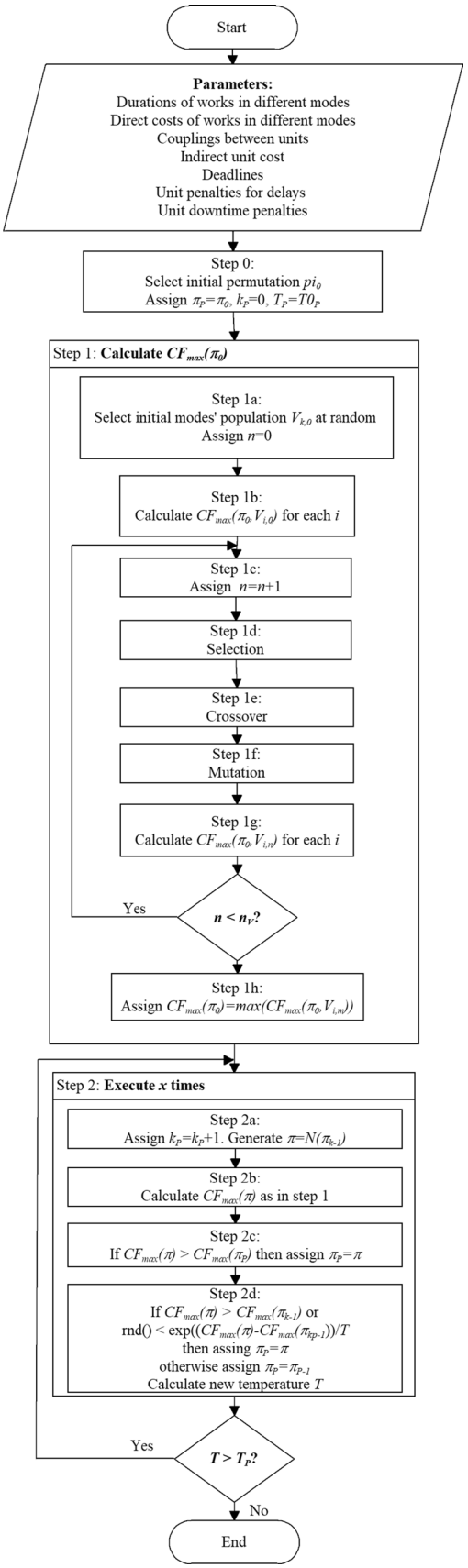

Fig. 5 SA + GS algorithm block diagram

of profit maximization is solved suboptimally, in accordance with the relationships presented in chapter 3, using the simulated annealing algorithm or the genetic search algorithm.

4.1 Simulated annealing algorithm

Simulated annealing (SA) algorithm has been proposed in the work of Kirkpatrick [40]. This algorithm uses analogous to the thermodynamic process of cooling the solid to introduce the trajectory of the search of the local extremum. States of solid matter are seen analogously as individual solution to the problem, whereas the energy of the body as the value of the objective function. During the physical process of cooling, the temperature is reduced slowly to maintain energy balance. The SA algorithm starts with the initial solution (usually chosen at random). Then, in each iteration, according to established rules or randomly, there is solution π' selected from the base neighborhood π. It becomes the base solution in the next iteration, if the value of the objective function is better than the current base solution or if it otherwise may become the base solution with the probability of

$$p = \exp(-D/T_i),$$

where $\pi = c(\pi) - c(\pi')$, T_i is the temperature of the current iteration i, c is the objective function. In each iteration, there are m draws from the neighborhood of the current basic solution performed. The parameter called the temperature decreases in the same way as in the natural process of annealing. The most frequently adopted patterns of cooling are geometrical or logarithmic ones. Below is the general algorithm of the SA method used to solve flow problems, which will be used to solve the problem under consideration.

SA algorithms are used to solve many optimization problems, including flow shop problems which are considered in the context of discrete optimization problems.

4.2 Genetic search algorithm

Genetic search algorithms (GS) use the principles of evolution in nature, which lead to the best adaptation (optimization) of individuals to the conditions found in a given environment. GS algorithm concept was presented in [41, 42]. They use a population of individuals (solutions), which are then processed during the selection, actions induced by the use of genetic operators, and the survival phase. The population in GS algorithms is a set of individuals representing solutions. Each solution is coded by a set of attributes stored in the genetic material (chromosomes, genes). There are many coding methods specific for various optimization problems, e.g., for flow shop problems, subsequent solutions are coded in the chromosome directly using permutation [43]. The population is processed by means of cyclically consecutive processes: reproduction, crossing, and mutation as well as survival or selection. In the reproduction phase, individuals are reproduced in proportion to the measure of adaptation to the environment. The adaptation function, which is a measure of adaptation, can be, e.g., the value of the objective function for a given solution. This process means that individuals with better adaptation will have more descendants in the next generations. Individuals selected from the population form the so-called parental pool from which the pairs are selected (so-called parents) providing individuals of a new generation. They are created using the genetic crossing operator. Then, the mutation process is carried out causing changes in the genetic material, which usually occur with a low probability

Algorithm SA

Step 0. Determine the initial solution $\pi \in \Pi$. Substitute $\pi_{SA} = \pi^0$, $k = 0$, $T = T_0$.

Step 1. Perform steps 1.1 – 1.3 x times.

Step 1.1. Substitute $k := k + 1$. Choose at random $\pi \in N(V, \pi^{k-1})$.

Step 1.2. If $c(\pi) > c(\pi_{SA})$ then substitute $\pi_{SA} = \pi$.

Step 1.3. If $c(\pi) > c(\pi_{k-1})$ then substitute $\pi_k = \pi$.

Otherwise, accept solution π with a probability of $p = \exp((c(\pi) - c(\pi_{k-1}))/T$, where $\pi_k = \pi'$.

If solution π was accepted, and $\pi_k = \pi_{k-1}$ if solution π was not accepted.

Step 2. Change the temperature T according to the defined pattern of cooling.

Step 3. If $T > T_N$ return to the step 1, otherwise STOP.

allowing for the slow introduction of innovation in a generation. In the survival (selection) phase, individuals are selected that will form part of the new population.

Selection can be carried out, e.g., in accordance with the roulette principle (rank selection), in which better chances are given to better-adapted individuals or on a tournament basis. The conditions for stopping the algorithm can be: time limit, the maximum number of iterations, achieving a satisfactory value of the objective function, or the optimal value. The GS method contains many elements that can be freely defined: chromosome coding method, crossover and mutation operators, adaptation function, parent pool selection, and parent matching scheme, survival scheme. The following is the general algorithm for the GS method:

Algorithm GS

Step 0. Generate initial population $P(0)$, $t = 0$, assess $P(0)$.

Step 1. $t := t + 1$; Choose the most suited individuals (selection).

Step 2. Change selected individuals (genetic transformation: crossing, mutation).

Step 3. Assess $P(t)$. Delete the weakest individuals (survival phase).

Step 4. If the stop condition is met, STOP, otherwise return to step 1.

GS algorithms are currently successfully used to solve a number of optimization problems, including those related to the theory of task scheduling [43]. The GS method has imperfections that can manifest, mainly early convergence to the local extreme or poor convergence to optimal solutions or close to them. However, it behaves quite well for examples of small sizes, such as the model presented in the article.

4.3 The SA + SA form of solving the problem according to the presented model

The method of solving the optimization task in the presented model is consistent with the SA algorithm steps. When searching the set of possible sequencing (permutations) of object execution (the first decision variable), the following assumptions were made regarding its parameters:

- the $N(\pi)$ environment contains permutations generated from π using the "replace" motion,
- Boltzmann acceptance function was used,
- the geometric cooling scheme was adopted, i.e., $T_{i+1} = \lambda T_i$ and the initial temperature $T_0 = 60$, $\lambda = 0.7$, the number of solutions considered at the set temperature: 2, the minimum temperature $T_{\min} = 0.01$.

While calculating the value of the objective function, for a given permutation π, the task of optimizing profit maximization is solved, taking into account the second decision variable, i.e., a set of numbers presenting ways of performing works (modes). For this stage, the following assumptions were made regarding the parameters of the SA algorithm used in it:

- the neighborhood of a given set of numbers presenting the ways of performing works contains sets generated from the output set by means of the "change to another random" movement,
- Boltzmann acceptance function was used,
- the geometric cooling scheme was adopted, i.e., $T_{i+1} = \lambda T_i$ and the initial temperature $T_0 = 60$, $\lambda = 0.99$, the number of solutions considered at the set temperature: 6, the mum temperature $T_{\min} = 0.01$.

The presented approach using the SA algorithm for both decision variables will be referred to as the SA + SA algorithm in the following part of the article and is presented in the form of a block diagram in Fig. 4. The SA + SA algorithm has been implemented in the Python programming language.

4.4 The SA + GS form of solving the problem according to the presented model

Similarly to the SA + SA algorithm, the method of solving the optimization task in the presented model is consistent with the SA algorithm steps. While searching the set of possible ordering (permutations) of object execution (the first decision variable), assumptions regarding its parameters were made the same as in the SA + SA algorithm presented in chapter 4.3.

In contrast to the SA + SA algorithm, the GS algorithm was used only when calculating the value of the objective function, when for a given permutation π, the task of

optimizing profit maximization is solved taking into account the second decision variable, i.e., a set of numbers presenting the ways (modes) of performing works. For the first decision variable, i.e., the order of execution of the units (objects), it is assumed that the SA algorithm will still be used. Further in the article, the algorithm name SA + GS will be used for this combination of algorithms. The following assumptions were made regarding the form and parameters of the GS algorithm in the SA + GS algorithm:

- a randomly created population consists of individuals—sets of numbers representing methods of works implementation R, its size is equal to 90,
- selection—parents are selected using tournament selection, 40% of individuals are selected,
- crossing—the single point crossing operator was used, parents are chosen randomly,
- mutation is implemented on individuals of the generation by means of the "change by 1" motion applied to a randomly selected individual from the generation, part of the population is subject to mutation: 5%, probability of mutation: 0.01.
- the maximum number of iterations of the GS algorithm is 90.

The SA + GS algorithm has also been implemented in the Python programming language and is presented in the form of a block diagram in Fig. 5.

5 Verification of the results

The authors verified the results obtained with the methods presented above. For this purpose, three groups of examples of objects were generated randomly: $n = 6$ units with $m = 2$ works, $n = 5$ units with $m = 3$ works, and $n = 4$ units with $m = 4$ works. Such sizes are characteristic for small problems in the practice of multi-unit construction projects. There were five examples generated in each group. In each of the examples, a maximum value of the total profit of a

Table 1 Outcomes of the verification of the results for small examples obtained with the use of the SA + SA and SA + GS algorithms

Example name: number of units, x number of works_example number	The average value of the total profit TP_{SA+SA} obtained with the use of SA + SA algorithm	The average value of the total profit TP_{SA+GS} obtained with the use of SA + GS algorithm	The maximum value of the total profit TP_{ES} obtained with the use of ES algorithm	Average PRD(SA + SA) [%]	Average PRD(SA + GS) [%]
Examples $n = 6$ units, $m = 2$ works					
6×2_1	1031.34	1032.13	1032.21	0.085	0.009
6×2_2	1059.69	1059.82	1059.97	0.027	0.015
6×2_3	1038.84	1038.84	1039.20	0.035	0.035
6×2_4	1118.25	1116.01	1119.95	0.152	0.352
6×2_5	1054.26	1054.91	1055.07	0.077	0.015
Average *PRD*(SA + SA) for size 6×2 [%]:				0.075	
Average *PRD*(SA + GS) for size 6×2 [%]:					0.085
Examples $n = 5$ units, $m = 3$ works					
5×3_1	1217.61	1221.30	1224.32	0.548	0.247
5×3_2	1160.04	1172.11	1177.23	1.461	0.435
5×3_3	1208.81	1210.57	1213.51	0.387	0.243
5×3_4	1201.89	1201.65	1208.23	0.525	0.544
5×3_5	1173.78	1178.10	1180.61	0.578	0.212
Average PRD(SA + SA) for size 5×3 [%]:				0.700	
Average PRD(SA + GS) for size 5×3 [%]:					0.336
Examples $n = 4$ units, $m = 4$ works					
4×4_1	1280.81	1282.43	1315.84	2.662	2.539
4×4_2	970.89	995.48	1038.38	6.500	4.131
4×4_3	822.87	837.83	884.69	6.988	5.297
4×4_4	1322.12	1324.70	1352.63	2.256	2.065
4×4_5	1207.36	1215.29	1242.14	2.801	2.162
Average PRD(SA + SA) for size 4×4 [%]:				4.241	
Average PRD(SA + GS) for size 4×4 [%]:					3.239
Average PRD(SA + SA) [%]:				1.672	
Average PRD(SA + GS) [%]:					1.220

given project was sought in accordance with the presented model. Each of the studied examples was resolved five times. The results obtained with the help of the original software are presented in Table 1. Then, the three groups of analyzed examples were solved optimally by means of the exhaustive search (ES) algorithm. The obtained results were compared with each other by calculating the percentage relative difference *PRD(*SA + SA*)* of the SA + SA algorithm and *PRD(*SA + GS*)* of the SA + GS algorithms:

$$\mathrm{PRD(SA+SA)} = 100\%\frac{K^{\mathrm{ES}} - K^{\mathrm{SA+SA}}}{K^{\mathrm{ES}}} \tag{12}$$

$$\mathrm{PRD(SA+GS)} = 100\%\frac{K^{\mathrm{ES}} - K^{\mathrm{SA+GS}}}{K^{\mathrm{ES}}}, \tag{13}$$

where $K^{\mathrm{SA+SA}}$ is the value of the adopted objective function obtained by means of the SA + SA algorithm, $K^{\mathrm{SA+GS}}$ is the value of the adopted objective function obtained by means of the SA + GS algorithm, and K^{ES} is the value of the assumed objective function obtained by means of an exhaustive search algorithm.

The average PRD of the applied algorithms are presented in Table 1. The values of these differences are quite small, which confirms the high effectiveness of the algorithms in searching for optimal solutions in the subject. However, the results obtained by the SA + GS algorithm for small examples were slightly better with a average PRD difference of 0.452.

Additional tests were conducted for the bigger examples with $n = 10$, 15, 20, and $m = 3$, 5, 7. There were five examples generated for each size. Each of the studied examples was resolved five times. The number of examples for each of the considered sizes (in total 60 examples for all sizes) and the number of attempts to solve them is sufficient to compile statistical data, and draw conclusions about the quality of the results achieved by the tested algorithms. The average results obtained with the help of the created software are presented in Table 2. These examples were too big to be calculated by ES algorithm in a reasonable time, so the results obtained by SA + SA and SA + GS algorithms were tested using random search algorithm (RS algorithm) [44]. *PRD(*SA + SA*)* of the SA + SA algorithm and *PRD(*SA + GS*)* of the SA + GS algorithms were calculated as follows:

$$\mathrm{PRD(SA+SA)} = 100\%\frac{K^{\mathrm{RS}} - K^{\mathrm{SA+SA}}}{K^{\mathrm{RS}}} \tag{14}$$

$$\mathrm{PRD(SA+GS)} = 100\%\frac{K^{\mathrm{RS}} - K^{\mathrm{SA+GS}}}{K^{\mathrm{RS}}}, \tag{15}$$

where K^{RS} is the value of the adopted objective function obtained by means of the RS algorithm.

The average percentage relative differences of the applied algorithms for bigger examples are presented in Table 2. In the conducted experiments, for sizes, $n = 10$, 15, 20, and $m = 3$, 5, 7 the results of the SA + GS algorithm were always better (average PRD of − 8.02%) than the results of the SA + SA algorithm (average PRD of − 1.46%). That is why SA + GS algorithm was selected for the analysis of real-life problem instances. The results of the SA + GS and SA + SA algorithms were always better than the results of the RS algorithm.

6 Case study

The contractor, at the request of the investor, is to carry out a project consisting in the construction of $n = 5$ residential buildings (units). Each of them requires execution of $m = 5$ works carried out in a fixed order. Technological and organizational limitation ensures that a work cannot start if the work of the same type in the previous building did not end and if the previous work in the same building was not completed. The project will be implemented in full by the subcontractors. The contractor received numerous bids from the subcontractors which results in a possibility of executing each type of work in one of the three different variants (modes). Each mode has a fixed execution time of a given type of work (expressed in working days) and implementation cost (expressed in EUR). Execution modes are presented in Table 3.

Table 2 Average outcomes of the verification of the results for examples $n = 10$, 15, 20 obtained with the use of the SA + SA and SA + GS algorithms

Example name: number of units (*n*), *x* number of works (*m*)	Average PRD (SA + SA) [%]	Average PRD (SA + GS) [%]
Examples $n = 10$ units		
10 × 3	− 1.13	− 3.34
10 × 5	− 1.69	− 7.96
10 × 7	− 2.62	− 12.77
Average PRD for size $n = 10$	− 1.81	− 8.03
Examples $n = 15$ units		
15 × 3	− 0.92	− 4.46
15 × 5	− 1.09	− 7.07
15 × 7	− 2.11	− 12.71
Average PRD for size $n = 15$	− 1.37	− 8.08
Examples $n = 20$ units		
20 × 3	− 0.96	− 4.97
20 × 5	− 1.05	− 7.71
20 × 7	− 1.58	− 11.17
Average PRD for size n = 20	− 1.20	− 7.95
All examples		
Average PRD for all sizes [%]	− 1.46	− 8.02

Table 3 Modes of the works execution in $n=5$ buildings for $m=5$ works including the duration of the works and their cost

Modes (number/details)		Units $i=$				
		1	2	3	4	5
Earthworks and foundations ($j=1$)						
1	Execution time [working days]	15	18	18	12	21
	Execution cost [in thous. EUR]	12.95	15.97	12.06	15.66	14.54
2	Execution time [working days]	11	15	12	11	16
	Execution cost [in thous. EUR]	16.19	21.29	13.55	22.7	15.31
3	Execution time [working days]	9	10	8	7	11
	Execution cost [in thous. EUR]	20.56	26.61	17.07	24.97	16.53
Structural works ($j=2$)						
1	Execution time [working days]	66	81	77	55	91
	Execution cost [in thous. EUR]	85.67	91.32	60.62	64.7	87.93
2	Execution time [working days]	55	65	62	40	71
	Execution cost [in thous. EUR]	133.86	120.16	85.38	98.03	102.24
3	Execution time [working days]	50	46	47	27	60
	Execution cost [in thous. EUR]	157.96	129.77	92.21	114.69	108.38
Plumbing, HVAC, electrical works ($j=3$)						
1	Execution time [working days]	19	21	29	31	20
	Execution cost [in thous. EUR]	26.5	40.04	40.33	47.93	34.92
2	Execution time [working days]	16	18	21	24	19
	Execution cost [in thous. EUR]	33.13	46.56	52.38	57.75	53.72
3	Execution time [working days]	10	13	14	19	14
	Execution cost [in thous. EUR]	46.38	50.75	63.9	79.12	68.76
Internal finishing ($j=4$)						
1	Execution time [working days]	61	73	74	62	55
	Execution cost [in thous. EUR]	38.74	33.08	47.76	44.26	59.55
2	Execution time [working days]	53	69	65	46	42
	Execution cost [in thous. EUR]	58.7	53.36	62.03	52.69	64.03
3	Execution time [working days]	44	55	42	33	28
	Execution cost [in thous. EUR]	61.63	61.36	70.1	72.72	82.6
External finishing ($j=5$)						
1	Execution time [working days]	30	36	39	34	24
	Execution cost [in thous. EUR]	22.32	20.05	22.33	18.05	24.61
2	Execution time [working days]	26	25	29	25	17
	Execution cost [in thous. EUR]	30.58	25.7	33.83	23.75	27.65
3	Execution time [working days]	22	16	18	23	10
	Execution cost [in thous. EUR]	36.09	30.58	43.31	33.01	36.78

The calculation example was created on the basis of a model residential building (determined on the basis of the price catalogue), for which the duration of works and their direct costs were determined. On the basis of these data, the authors generated: the duration of works, and the direct cost of the object, the so-called medium building. The time and direct cost of each job were modified by a random number from − 30 to 30%. The remaining objects were generated on the basis of the so-called medium building using the uniform distribution. When creating the data on the time and cost of works, the principle was followed that any reduction in the duration of an activity resulted in an increase in its cost. The example was generated using randomness due to the inability to obtain real data for such an undertaking with sizes $n=5$ objects and $m=5$ works. For computation purposes, the authors used: Intel Core i5-4440, 4 GB RAM, OS Windows 10, Python 3.6 (with the uniform distribution based on numpy.random.randint function).

Between the works realized in the technological order, there are couplings between units that have been established on the basis of existing technological constraints which are shown in Table 4.

The duration of a billing period is equal to 20 working days which corresponds with one working month. This duration is set in accordance with construction practice in which works are invoiced on a monthly basis. Negative monthly

cash flows force the contractor to take loans at 9% per annum (paid off every month). The indirect costs incurred by the contractor (related to running and supervising the construction) are estimated to be 730 € for each day of the project implementation. The production value is subject to discount each month at the level of 8% per annum. The contractor assumed profit at the level of 12%.

The contractor has a deadline for the implementation of individual buildings imposed by the investor. The directive deadlines are 150, 210, 270, 350, and 380 (in working days) for subsequent units (objects). For failure to meet deadlines, penalties are imposed for the general contractor for each day of delay. For each building, the penalties are respectively: 540, 490, 580, 590, and 570 €. The penalties for discontinuities of work crews have been determined with subcontractors and amount respectively for each type of work (j from 1 to 5) to 200, 200, 100, 0, and 300 € for each day of stoppage. Payment and penalties delays are both equal to 20 business days.

The number of possible solutions for this problem instance is $5!\times3^{5\times5} \approx 1.02\times10^{14}$. The optimization goal for the contractor is to maximize the total profit—*TP* (cumulated cash flow for the last month of the project). For the initial schedule of $\pi_0=(1,2,3,4,5)$ and assumed times and costs of execution of all works in mode 2. Total profit of 143.87 thousand of euro was obtained. The duration of the entire project is at the level of 351 days. However, not all the deadlines for units were met and some downtime penalties were charged. The schedule for the initial $\pi^0=(1,2,3,4,5)$ rank without optimization is shown in Fig. 6a. The cumulated cash flow for this schedule is presented in Fig. 7a.

Table 4 Couplings between units s occurring between the works j and $j+1$ for $n=5$ units

Works $j=$	Value s [working days]				
	Units $i=$				
	1	2	3	4	5
1	5	5	5	5	5
2	− 5	− 5	− 5	− 5	− 5
3	− 5	− 5	− 5	− 5	− 5
4	− 10	− 10	− 10	− 10	− 10

The next step in the search for an optimal solution was the use of the SA + GS algorithm to find the optimal schedule of the project that maximizes the assumed goal function, taking into account the possible changes in the order of construction objects execution, and selected execution modes. The algorithm performed 51 iterations according to the parameters described in Sect. 3.

The biggest *TP* calculated for the project is 204.58 thousand Euro obtained for the following scheduling of units: $\pi_{SA+GS}=(2,3,5,1,4)$. The selected execution modes are presented in Table 5. The deadline for the project implementation was 283 days. In relation to the initial schedule π_0, the total profit was improved by 42% (60.71 thousand Euro) and the deadline for implementation was improved by 19% (68 days). The obtained schedule is shown in Fig. 6b. The cumulated cash flow for this schedule is presented in Fig. 7b.

7 Summary and conclusions

Scheduling of multi-unit construction projects, which is a case of repetitive projects, is a problem in which discrete optimization tasks often occur. These problems are usually NP-hard due to the fact that they may be qualified as the permutation flow shop problems. In the presented model

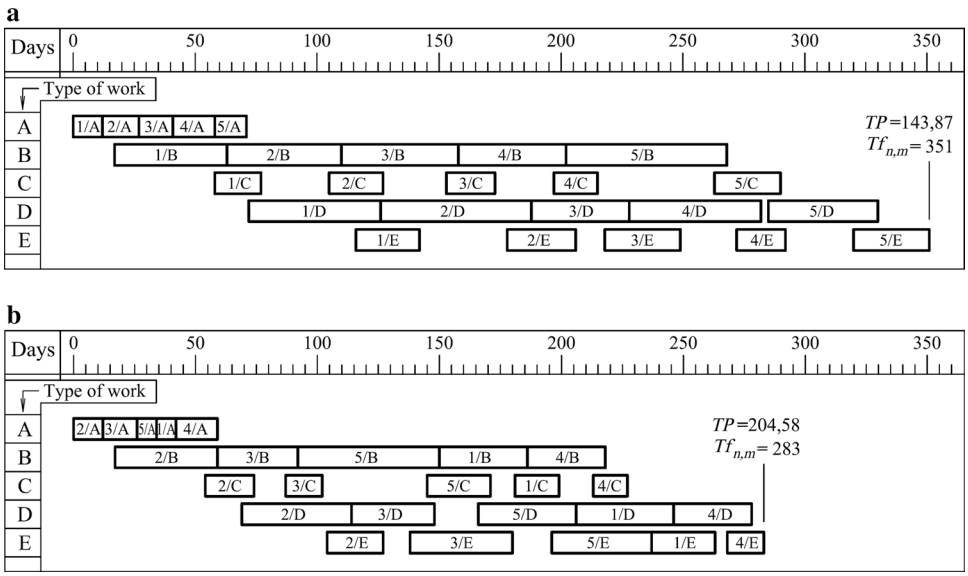

Fig. 6 **a** Schedule for the implementation of project in the case study for π^0. **b** Schedule for the implementation of project in the case study for π_{SA+GS} with optimization using SA + GS algorithm

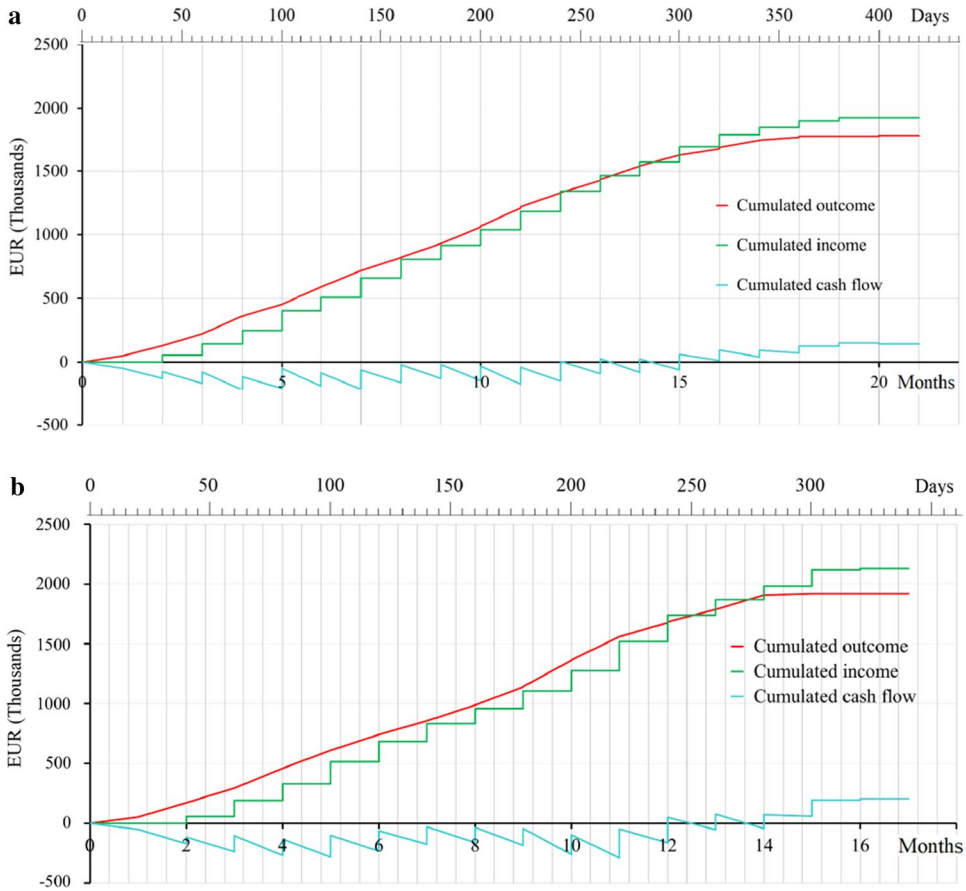

Fig. 7 **a** Cumulated cash flows for the implementation of project in the case study for π^0. **b** Cumulated cash flows for the implementation of project in the case study for π_{SA+GS} with optimization using SA + GS algorithm

Table 5 Best execution modes for $\pi_{SA+GS} = (2, 3, 5, 1, 4)$ with optimization using SA + GS algorithm

Modes	Units $i=$				
	1	2	3	4	5
Earthworks and foundations ($j=1$)					
Selected modes k	3	3	2	2	3
Structural works ($j=2$)					
Selected modes k	3	3	3	3	3
Plumbing, HVAC, electrical works ($j=3$)					
Selected modes k	3	3	3	3	3
Internal finishing ($j=4$)					
Selected modes k	3	3	3	3	3
External finishing ($j=5$)					
Selected modes k	2	3	1	3	1

application of the additional parameter namely, the selection of the works' execution mode resulted in a significant increase in the number of possible solutions. Consequently, the analyzed discrete optimization problem had two different decision variables. The developed model considered monthly cash flows subject to direct and indirect costs, penalties for missing deadlines, costs of work group discontinuities, and borrowing losses. The latter factor is often overlooked in research. Meanwhile, it can have a significant impact on the outcome of projects. Especially in construction projects in which payment delays and contractual penalties occur (both aspects were included in the model).

Two approaches were tested to solve the problem, a combination of two SA algorithms, and SA algorithm supported by GS algorithm. The results obtained using the second combination were slightly better than results obtained using only SA algorithms. Due to the lack of research in the literature regarding the applied form of the algorithms, the article presents verification of the results provided with its use. It showed its high effectiveness in searching for optimal solutions in the model in question.

The presented model of multi-unit construction project scheduling can be used when determining the optimal work schedule for construction companies using the flow organization system of work. It can be used in the situation when companies use their own working crews to implement the works or intend to select specialized subcontractors who best meet the imposed conditions. The developed model can be easily implemented in scheduling programs and can support the planner when designing multi-unit construction projects, helping to maximize the profit of the planned undertaking.

Due to the development of dedicated software for solving optimization tasks, it is possible to extend the current model with additional technological and organizational constraints, additional factors affecting the cost of the project and taking into account other objective functions.

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

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Zhou J, Love PE, Wang X, Teo KL, Irani Z. A review of methods and algorithms for optimizing construction scheduling. J Oper Res Soc. 2013. https://doi.org/10.1057/jors.2012.174.
2. Herreolen W. Project scheduling—theory and practice. Prod Oper Manag. 2005. https://doi.org/10.1111/j.1937-5956.2005.tb00230.x.
3. Węglarz J, et al. Project scheduling—recent models, algorithms and applications. Springer; 2012.
4. Kolisch R, Padman R. An integrated survey of deterministic project scheduling. Omega. 2001. https://doi.org/10.1016/S0305-0483(00)00046-3.
5. Herroelen W, De Reyck B, Demeulemeester E. Resource-constrained project scheduling: a survey of recent developments. Comput Oper Res. 1998. https://doi.org/10.1016/S0305-0548(97)00055-5.
6. Brucker P, Drexl A, Mohring R, Neumann K, Pesch E. Resource-constrained project scheduling: Notation, classification, models, and methods. Eur J Oper Res. 1999. https://doi.org/10.1016/S0377-2217(98)00204-5.
7. Habibi F, Barzinpour F, Sadjadi S. Resource-constrained project scheduling problem: review of past and recent developments. J Proj Manag. 2018. https://doi.org/10.5267/j.jpm.2018.1.005.
8. Van Peteghem V, Vanhoucke M. A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem. Eur J Oper Res. 2010. https://doi.org/10.1016/j.ejor.2009.03.034.
9. Rosłon J. The multi-mode, resource-constrained project scheduling problem in construction: state of art review and research challenges. Tech Trans. 2017. https://doi.org/10.4467/2353737XCT.17.070.6427.
10. Senouci AB, Eldin NN. Use of genetic algorithms in resource scheduling of construction projects. J Constr Eng Manag. 2004. https://doi.org/10.1061/(ASCE)0733-9364(2004)130:6(869).
11. Mika M, Waligóra G, Węglarz J. Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models. Eur J Oper Res. 2005. https://doi.org/10.1016/j.ejor.2003.10.053.
12. Tiwari V, Patterson JH, Mabert VA. Scheduling projects with heterogeneous resources to meet time and quality objectives. Eur J Oper Res. 2009. https://doi.org/10.1016/j.ejor.2007.11.005.
13. Ghoddousi P, Eshtehardian E, Jooybanpour S, Javanmardi A. Multi-mode resource-constrained discrete time–cost-resource optimization in project scheduling using non-dominated sorting genetic algorithm. Autom Constr. 2013. https://doi.org/10.1016/j.autcon.2012.11.014.
14. Geiger MJ. A multi-threaded local search algorithm and computer implementation for the multi-mode, resource-constrained multi-project scheduling problem. Eur J Oper Res. 2017. https://doi.org/10.1016/j.ejor.2016.07.024.
15. Rosłon J, Kulejewski J. A hybrid approach for solving multi-mode resource-constrained project scheduling problem in construction. Open Engineering. 2019. https://doi.org/10.1515/eng-2019-0006.
16. Deblaere F, Demeulemeester E, Herroelen W. Reactive scheduling in the multi-mode RCPSP. Comput Oper Res. 2011. https://doi.org/10.1016/j.cor.2010.01.001.
17. Chen A, Liang YC, Padilla J. An entropy-based upper bound methodology for robust predictive multi-mode RCPSP schedules. Entropy. 2014. https://doi.org/10.3390/e16095032.
18. Liao TW, Egbelu PJ, Sarker BR, Leu SS. Metaheuristics for project and construction management—a state-of-the-art review. Autom Constr. 2011. https://doi.org/10.1016/j.autcon.2010.12.006.
19. Mika M, Waligóra G, Węglarz J. Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times. Eur J Oper Res. 2008. https://doi.org/10.1016/j.ejor.2006.06.069.
20. Jarboui B, Damak N, Siarry P, Rebai A. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. Appl Math Comp. 2008;195:299–308.
21. Li H, Zhang H. Ant colony optimization-based multi-mode scheduling under renewable and nonrenewable resource constraints. Autom Constr. 2013. https://doi.org/10.1016/j.autcon.2013.05.030.
22. Sebt MH, Afshar MR, Alipouri YJEO. Hybridization of genetic algorithm and fully informed particle swarm for solving the multi-mode resource-constrained project scheduling problem. Eng Optim. 2017. https://doi.org/10.1080/0305215X.2016.1197610.
23. Zhang L, Luo Y, Zhang Y. Hybrid particle swarm and differential evolution algorithm for solving multimode resource-constrained project scheduling problem. J Control Sci Eng. 2015. https://doi.org/10.1155/2015/923791.
24. Özdamar L, Dündar H. A flexible heuristic for a multi-mode capital constrained project scheduling problem with probabilistic cash inflows. Comput Oper Res. 1997. https://doi.org/10.1016/S0305-0548(96)00058-5.
25. Chen PH, Weng H. A two-phase GA model for resource-constrained project scheduling. Autom Constr. 2009. https://doi.org/10.1016/j.autcon.2008.11.003.
26. Zhang Z, Xu J. A multi-mode resource-constrained project scheduling model with bi-random coefficients for drilling grouting construction project. Int J Civ Eng. 2013. https://doi.org/10.3934/jimo.2016.12.565.
27. Xu J, Feng C. Multimode resource-constrained multiple project scheduling problem under fuzzy random environment and its

application to a large scale hydropower construction project. Sci World J. 2014. https://doi.org/10.1155/2014/463692.
28. Hegazy T, Wassef N. Cost optimization in projects with repetitive nonserial activities. J Constr Eng Manag. 2001. https://doi.org/10.1061/(ASCE)0733-9364(2001)127:3(183).
29. Hegazy T, Elhakeem A, Elbeltagi E. Distributed scheduling model for infrastructure networks. J Constr Eng Manag. 2004. https://doi.org/10.1061/(ASCE)0733-9364(2004)130:2(160).
30. Zhang H, Li H, Tam CM. Heuristic scheduling of resource constrained, multiple mode and repetitive projects. Constr Manag Econ. 2006. https://doi.org/10.1080/01446190500184311.
31. Zhang L, Zou X. Repetitive project scheduling theory and methods. 1st ed. Elsevier; 2015.
32. Gupta J, Stafford EF Jr. Flowshop scheduling research after five decades. Eur J Oper Res. 2006. https://doi.org/10.1016/j.ejor.2005.02.001.
33. Bożejko W, Hejducki Z, Wodecki M. Applying metaheuristic strategies in construction projects management. J Civ Eng Manag. 2012. https://doi.org/10.3846/13923730.2012.719837.
34. Bożejko W, Hejducki Z, Uchroński M, Wodecki M. Solving resource-constrained construction scheduling problems with overlaps by metaheuristic. J Civ Eng Manag. 2014. https://doi.org/10.3846/13923730.2014.906496.
35. Bożejko W, Hejducki Z, Wodecki M. Flowshop scheduling of construction processes with uncertain parameters. Arch Civ Mech Eng. 2019. https://doi.org/10.1016/j.acme.2018.09.010.
36. Podolski M. Management of resources in multiunit construction projects with the use of a tabu search algorithm. J Civ Eng Manag. 2017. https://doi.org/10.3846/13923730.2015.1073616.
37. Podolski M, Sroka B. Cost optimization of multiunit construction projects using linear programming and metaheuristic-based simulated annealing algorithm. J Civ Eng Manag. 2019. https://doi.org/10.3846/jcem.2019.11308.
38. Zawistowski J, Kulejewski J. Influence of the contractor's payment method on the economic effectiveness of the construction project from the contractor's point of view. Open Eng. 2018. https://doi.org/10.1515/eng-2018-0055.
39. Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG. Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey. In: Proceedings of the advanced research institute on discrete optimization and systems applications of the systems science panel of NATO and of the Discrete Optimization Symposium. Elsevier. 1979.
40. Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. Science. 1983. https://doi.org/10.1126/science.220.4598.671.
41. Holland JH. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press; 1992.
42. Goldberg D. Genetic algorithms in search, optimization and machine learning. Addison Wesley; 1989.
43. Reeves CR. A genetic algorithm for flowshop sequencing. Comp Oper Res. 1995. https://doi.org/10.1016/0305-0548(93)E0014-K.
44. Karnopp DC. Random search techniques for optimization problems. Automatica. 1963. https://doi.org/10.1016/0005-1098(63)90018-9.