JOURNAL OF CIVIL ENGINEERING AND MANAGEMENT
ISSN 1392-3730 print/ISSN 1822-3605 online
2012 Volume 18(5): 621–630
doi:10.3846/13923730.2012.719837

APPLYING METAHEURISTIC STRATEGIES IN CONSTRUCTION PROJECTS MANAGEMENT

Wojciech Bożejko[1], **Zdzisław Hejducki**[2], **Mieczysław Wodecki**[3]
[1]*Wrocław University of Technology, Institute of Computer Engineering, Control and Robotics, Janiszewskiego 11-17, 50-372 Wrocław, Poland*
[2]*Wrocław University of Technology, Institute of Construction, Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland*
[3]*University of Wrocław, Institute of Computer Science, Joliot-Curie 15, 50-383 Wrocław, Poland*
E-mails: [1]wojciech.bozejko@pwr.wroc.pl (corresponding author); [2]zdzislaw.hejducki@pwr.wroc.pl; [3]mwd@ii.uni.wroc.pl

Received 18 Apr. 2012; accepted 02 May 2012

Abstract. This work deals with the application of artificial intelligence instruments in a building schedule. In this article there was presented an original optimization scatter search algorithm taking into consideration both technological and organizational restrictions. This algorithm was applied to the real analysis of the industrial building project realization.

Keywords: construction, strategy, metaheuristics, optimization, scheduling.

1. Introduction

Optimal scheduling of construction projects is an important task as it influences the effectiveness of executive companies in a crucial way. In a negotiation process of construction contracts there appears a necessity to define dates of project completion. It is a difficult problem as the level of uncertainty concerning different (Zavadskas *et al.* 2010a), constantly changing parameters is rather high. Not meeting the deadlines results in generating losses (contractual penalties or unused resources). This leads to a complex discrete-continuous optimization problems with irregular cost functions. While converting the issues of construction processes into a field of classical scheduling theory one may encounter many problems connected to choosing an appropriate model and an adequate algorithm. They are most commonly completely new, NP-hard problems of combinatorial optimization, which means that up to now there are no algorithms with polynomial computing time of solving it.

Industrial building is characterized with a necessity to execute works in a quick pace resulting from constraints connected to: for instance re-building certain objects during performing of production line processes, minimizing of downtimes on assembly lines, collisions of building works with production processes on a limited area of premises, etc. Realization of construction works requires coordination of many processes usually done by specialized executive companies. Short deadlines, competition, constraints in a transport field or storage of materials enable perfect planning and scheduling not only in a sphere directly connected to realization of construction works but also in its surroundings like: transport, ensuring continuity of team and equipment works, storage of materials and prefabrication components, access roads, etc. All the above mentioned elements are linked to one another or mutually dependant. Solving of each of the mentioned issues leads to complicated models and difficult (NP-hard) optimization tasks. It is even more important taking into consideration that in practice these tasks are multi-criteria ones and very often opposing to one another e.g. cost and time (shortening the deadline results in cost increasing). Moreover, dynamically changing environment (e.g. conditions for safe assembly, elimination of collisions in building process, satisfying technological conditions with reference to e.g. technological gaps – proper time for pouring concrete or organizational problems – ensuring stability of carrier system during assembly requires huge flexibility of computational methods supporting management. The process of planning and scheduling of works connected with construction of industrial buildings complex (presented below) is an example of such problem.

Linear Scheduling Methods (LSM) and Time Couplings Methods (TCM) stem from the construction work course, based on the project division into sectors. Nowadays they are used in building of motorways, waterworks, sewage systems, telephone lines, skyscrapers and other structures. Russell and Caselton (1988) introduce extensions to linear scheduling optimization having the form of interrupt duration vector choices. Russell and Wong (1993) describe the development and the use of a new generation of planning structures providing a basis for a superset for the traditional critical path method and over the years introduce various extensions needed to represent the realities of construction. Senouci and Eldin (1996) present a non-serial dynamic programming formulation for the scheduling of linear projects with non-sequential activities. El-Rayes and Moselhi (2001) optimize resources in repetitive construction projects. Harris and Ioannou (1998) work out a repetitive scheduling method with technical constraints. Adeli and Karim (1997) develop a neural dynamics cost optimization model for construction projects. Similar problems were also considered by Zavadskas *et al.* (2010b, 2011), Medineckiene *et al.* (2011), Burduk and Chlebus (2009), Nowakowski (2010), Zhang and Gao (2010), Liu and Wang (2011, 2012) and Chen *et al.* (2012). Pareto-optimal method was proposed by Jiang *et al.* (2011) for the construction project cash flow planning. The review of the modern approaches in this field can be found in the work of Sarker *et al.* (2012).

LSM and TCM can be also considered as the permutation flow shop scheduling problem described as follows. A number of tasks is to be processed on a number of machines. Each task must go through all the machines in exactly the same order and the tasks' order is the same on each machine. Each machine can process at most one task at any point in time, and each task can be processed on at most one machine at any time. The objective is to find a schedule that minimizes the completion time of the last task.

Johnson (1954) gives an $O(n\log n)$ algorithm for two-machines flow shop problems with C_{max} criterion and Garey *et al.* (1976) show that for 3 and more machines it is NP-hard. The best available branch and bound algorithms are these of Lageweg *et al.* (1978). However, their performance is not entirely satisfactory as they experience difficulty in solving instances with 20 tasks and 5 machines.

Various local search methods are available for the permutation flow shop problem. Tabu search algorithms are proposed by Glover and Laguna (1997) and improved by Grabowski and Wodecki (2004). Simulated annealing algorithms are proposed by Ishibushi *et al.* (1995). In our paper we present a sequential and parallel simulated annealing algorithm for the permutation flow shop problem.

2. Case study

In Kunice near Legnica (Poland) there began modernization works of ceramic materials production plant. Industrial halls were designed and then constructed on the factory premises to accommodate the production process, the tunnel furnace and the clay storehouse. Both the production and the storehouse halls were made from steel constructions covered with insulation panels of mineral wool and the Mega Plus roofing coat (see Figs 1, 2). Dehydration of the roof was conducted as gravitation one using Nicole system. The whole building area constituted about 20 000 m^2. The investor set a deadline for making the halls on the 20th of December because of the necessity to start the works concerning the tunnel furnace assembly. Scheduled works were connected with mutual relations (time coupling) which determined technological relations – between works there had to occur essential technological breaks as well as fixed sequence relations.

Works were done by different specialized building companies with which the general contractor signed individual contracts where the deadline dates of building works completion were set. Missing the deadline led to calculation of so-called stipulated penalties coming to 1 percent of the contract value for each day of delay (however, for each subcontractor rates were significantly different).

In planning of the works' realization a continuous way of doing different construction works was adopted. One of the authors participated in the building project realization. The building project consists in building of 3 halls where each hall building was composed of 5 stages. Tables 1–4 present stages names and the duration times (in days). The Fig. 3 shows the model schedule.

Optimization of realization process presented on the example of the construction works and models of works consists in a certain problem of tasks arrangement on machines, so-called permutational flow shop problem with a costs delay sum minimization. The optimal methods, concerning the NP-hard problems solution of

Fig. 1. The roof on the developed object Kunice

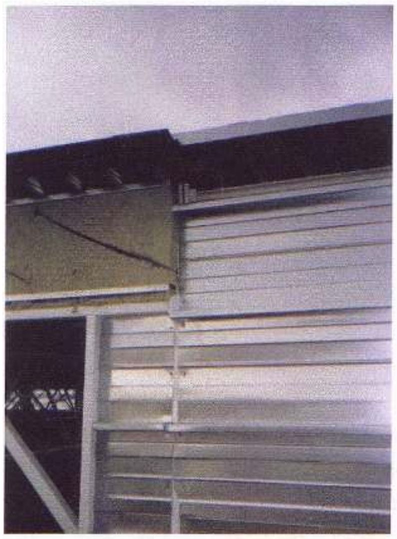
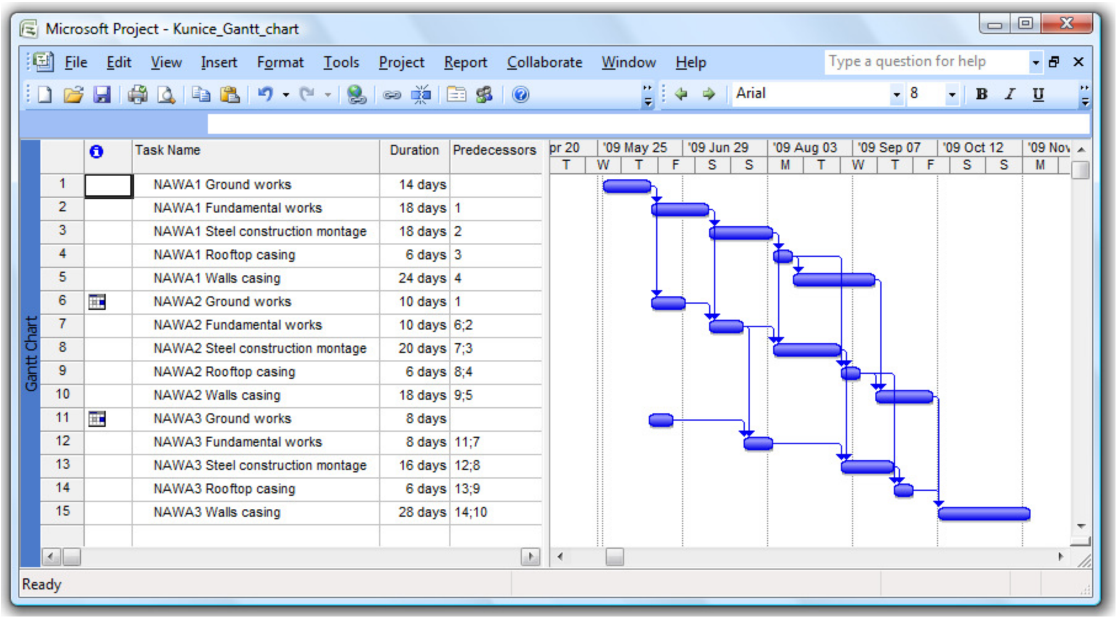


Fig. 2. Walls casing on the developed object Kunice


Microsoft Project - Kunice_Gantt_chart
File Edit View Insert Format Tools Project Report Collaborate Window Help
Type a question for help
Arial
8
Gantt Chart
pr 20
'09 May 25
'09 Jun 29
'09 Aug 03
'09 Sep 07
'09 Oct 12
'09 Nov
Ready

	Task Name	Duration	Predecessors
1	NAWA1 Ground works	14 days	
2	NAWA1 Fundamental works	18 days	1
3	NAWA1 Steel construction montage	18 days	2
4	NAWA1 Rooftop casing	6 days	3
5	NAWA1 Walls casing	24 days	4
6	NAWA2 Ground works	10 days	1
7	NAWA2 Fundamental works	10 days	6;2
8	NAWA2 Steel construction montage	20 days	7;3
9	NAWA2 Rooftop casing	6 days	8;4
10	NAWA2 Walls casing	18 days	9;5
11	NAWA3 Ground works	8 days	
12	NAWA3 Fundamental works	8 days	11;7
13	NAWA3 Steel construction montage	16 days	12;8
14	NAWA3 Rooftop casing	6 days	13;9
15	NAWA3 Walls casing	28 days	14;10

Fig. 3. The Gantt chart made in Microsoft Project for Kunice case study data

Table 1. Stages duration times (in days)

Hall	Stages				
	Ground works	Fundamental works	Steel Construction montage	Roof casing	Walls casing
NAWA1	14	18	18	6	24
NAWA2	10	10	20	6	18
NAWA3	8	8	16	6	28

Table 2. Stages deadlines times (in days)

Hall	Stages				
	Ground works	Fundamental works	Steel construction montage	Roof casing	Walls casing
NAWA1	14	30	52	54	78
NAWA2	26	50	68	78	100
NAWA3	32	50	80	90	130

Table 3. Stages contracts values (working days costs, in Euro)

Hall	Stages				
	Ground works	Fundamental works	Steel construction montage	Roof casing	Walls casing
NAWA1	22 400	43 200	28 800	19 200	67 200
NAWA2	16 000	24 000	32 000	19 200	50 400
NAWA3	12 800	19 200	25 600	19 200	78 400

Table 4. Penalties for each day of delay (from 1 to 2 percentage of the working days costs, in Euro)

Hall	Stages				
	Ground works	Fundamental works	Steel Construction montage	Roof casing	Walls casing
NAWA1	224	432	576	192	1008
NAWA2	160	240	640	192	756
NAWA3	128	192	512	192	1176

combinatorial optimization in regard of their exponential calculation time, are not widely applied in practice. It happens as a result of the fact that only instances of "small" size can be solved in a reasonable time, even the continuously increasing computer calculation power is not able to make the calculation time significantly shorter. Therefore, to solve such problems in practice, approximation algorithms are applied most commonly. They do not guarantee, however, that the determined algorithm is optimal. In literature, many different approximation algorithms are described: from the simplest ones usually being based on greedy procedures and sorting to very complex ones which find solutions only slightly different from optimal ones, on average even about less than 1 (for a certain group of reference data, see Grabowski, Wodecki 2004).

The most frequently used classic metaheuristics Tabu Search – TS, Simulated Annealing – AS and Genetic Algorithm – GA often finish their calculations in a certain local minimum. That's why hybrid algorithms connecting elements of different methods are used making the algorithms less sensitive to data specificity and "get stuck" in a certain local minimum. To solve the discussed in this work permutational flow shop problem we apply heuristic hybrid algorithm based on the Scatter Search method, along with the elements of TS, SA and GA methods.

3. The flow shop problem with the cost sum minimization

In many production processes we deal with a flow of manufactured elements in the technological course. They undergo a treatment operation on workplaces accommodated on "machines". In the production systems on time it is required that each element should be made before the fixed time completion. If the time is exceeded (delay), then penalty is calculated. The production process optimization consists in fixing such an order of elements manufacturing that it would provide us with optimal results measured by value of a certain criterion (e.g. a sum of penalties for a delay). It can be usually made by defining a combinatorial problem in which the optimal element should be determined (a sequence or a permutation) in a definite, usually very big set of acceptable solutions. Presented in the previous chapters issues connected with optimization of certain building projects' realization (in the language of scheduling tasks theory) can be defined as a problem of determining of the optimal sequence of tasks' realization on machines, and at the same time the costs delay sum of tasks constitutes the optimization criterion. Each task should be performed consecutively on all machines, but the sequence of tasks performance on each machine must be the same. In literature, it is known as a permutational flow shop problem with the costs delay sum minimization (Total Weighted Tardiness Flow Shop Problem, in brief *TWTFS*) and it is NP-hard (Lenstra *et al.* 1977) show that for a single machine case it is NP-hard). Nowadays, we don't know any attributes of *TWTFS* problem allowing us to provide with an indirect overview of some elements of a tasks set. Therefore, classical metaheuristics (searching with tabu, simulated annealing, genetic algorithm), which are very successful in solving many classical problems of tasks scheduling (mainly with a criterion $C_{\max}$, i.e. the minimization of all tasks' completion date (Grabowski, Wodecki 2004) aren't in this case much effective. In the further part of this work we present some aggregation tasks methods allowing us to eliminate many solutions which aren't perspective. The conducted calculation experiments proved that heuristic algorithm's efficiency improved significantly basing the scatter search method. Particularly, it's noticeable while solving instances with many numbers of tasks.

3.1. Problem definition

The Total Weighted Tardiness Flow Shop Problem (*TWTFS*) can be defined in the following way:

Problem: There is a set of n tasks $\mathsf{J}=\{1,2,\ldots,n\}$ which are to be performed by means of m machines from the set $\mathsf{P}=\{1,2,\ldots,m\}$. The task $j\in\mathsf{J}$ should be performed consecutively on each machine, but the task performance on k machine (operation O_{jk}) can start only after the completion of this task performance on a machine $\boldsymbol{k-1}$ $(\boldsymbol{k=2,3,\ldots,m})$ and the sequence of tasks' performance on each machine must be the same. Furthermore, in any moment the machine can perform at most one task and the task's performance cannot be broken. For the task $i\in\mathsf{J}$, let p_{ij} be the performance time on the machine $j\in\mathsf{P}$, d_i the required completion time, and w_i the cost function weight.

For the set tasks sequence by $C_{i,j}$ we mark the completion time of the task $i\in\mathsf{J}$ on a machine $j\in\mathsf{P}$. Then, $T_i=\max\{0,C_{i,m}-d_i\}$ is a tardiness, and $f_i(C_{i,m})=w_i\cdot T_i$ cost of tardiness of task performance (see Fig. 4). It should determine such a sequence of tasks performance that it would minimize the costs tardiness sum i.e. the sum $\sum_{i=1}^{n}f_i(C_{i,m})=\sum_{i=1}^{n}w_i\cdot T_i$.

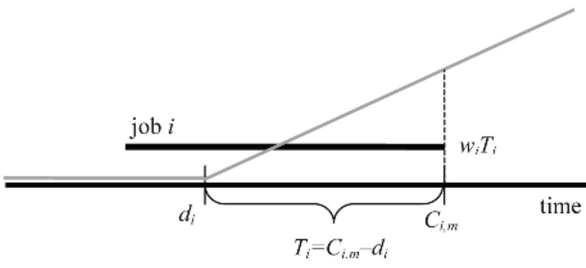


Fig. 4. Tardiness function

Problems of tasks' scheduling with sum-cost goal functions have a long, over thirty-year history. There is a great number of works concerning scheduling on one machine. Research for multi-machine problems more often involves the issue of delay of tasks number minimization, i.e. $\sum U_i$ (where $U_i=1$, if $C_{i,m}>d_i$, otherwise $U_i=0$). In the work of Lenstra *et al.* (1977) it is proved that the flow shop problem for two machines with $\sum U_i$ criterion is NP-hard. To the most important works containing the main theoretical results as well as good approximation algorithms belongs the Bulfin and M'Hallah (2003).

For the flow shop problem with many machines and $\sum U_i$ criterion Hariri and Potts (1989) described an optimal algorithm based on the division and restriction method, by means of which in a reasonable time instances with 25 tasks and 3 machines can be solved.

Genetic algorithms presented in works of Ucar and Tasgetiren (2006) are ones of a few metaheuristic algorithms which have been made exclusively to solve multi-machine flow shop problem with a criterion $\sum w_i U_i$. There are considerably less works dedicated to problems of costs delay sum minimization, i.e. the function $\sum w_i T_i$ and its particular example $\sum T_i$. The difficulty of the problem problem is verified by the fact that for the simplest example of two machines and goal function $\sum T_i$ the newest exact algorithms (Schaller 2005) solve instances to 20 tasks, whereas for the most difficult (discussed in this section) flow shop problem with $\sum w_i T_i$ criterion only a few works have been published describing optimal algorithms (the newest one – Chung *et al.* 2006).

There are also works dedicated to approximation algorithms. Based on different methods the newest heuristics and metaheuristics (mainly evolutionary and stimulated annealing) are presented in works of Hasija and Rajendran (2004) and Onwubolu and Davendra (2006). A wide range of methods, algorithms and publications covering multi-machine problems of tasks scheduling with costs delay sum minimization is included in works of Valada *et al.* (2006).

At first we shall introduce necessary definitions and notations as well as describe aggregation tasks methods (block construction) significantly improving the efficiency of algorithms. The calculation experiments were carried out on examples generated randomly using the appropriately modified method presented in the Taillard's work (Taillard 1993).

Each problem *TWTFS* solution can be represented by a permutation $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$ of tasks from the J set. By Φ_n we mark a set of all permutations of the J set of elements. Let $S_{i,j}$ be the *starting* time of a task $i \in \mathsf{J}$ on a machine $j \in \mathsf{P}$. For any permutation $\pi \in \Phi_n$ times of tasks starting on particular machines can be determined from the following recursive relations:

$$S_{\pi(1),1} = 0, \quad S_{\pi(i),1} = S_{\pi(i-1),1} + p_{\pi(i-1),1}, \quad i = 2,3,\ldots,n, \tag{1}$$

$$S_{\pi(1),j} = S_{\pi(1),j-1} + p_{\pi(1),j-1}, \quad j = 2,3,\ldots,m, \tag{2}$$

$$S_{\pi(i),j} = \max\{S_{\pi(i),j-1} + p_{\pi(i),j-1}, S_{\pi(i-1),j} + p_{\pi(i-1),j}\} \tag{3}$$

for $i = 2,3,\ldots,n,\ j = 2,3,\ldots,m$. In this case the times of task's completion:

$$C_{\pi(i),j} = S_{\pi(i)j} + p_{\pi(i),j}, i = 1,2,\ldots,n,\ j = 1,2,\ldots,m. \tag{4}$$

By $T_{\pi(i)} = \max\{C_{\pi(i),m} - d_{\pi(i)}, 0\}$ we mark the task delay, where $C_{\pi(i),m}$ is the completion time $\pi(i)$ on the last machine. If $T_{\pi(i)} > 0$, then the task is *late*, and on the contrary the case *early*. The product of $w_{\pi(i)} \cdot T_{\pi(i)}$ we call the *cost execution* (a penalty for late) of the task $\pi(i)$, whereas:

$$\mathsf{WT}(\pi) = \sum_{i=1}^{n} w_{\pi(i)} \cdot T_{\pi(i)}, \tag{5}$$

is the π permutation *cost* (weight). The *TWTFS* problem can be represented as an optimal permutation $\pi^* \in \Phi_n$ for which:

$$\mathsf{WT}(\pi^*) = \min\{\mathsf{WT}(\pi) : \pi \in \Phi_n\}. \tag{6}$$

To solve the *TWTFS* problem (to find the suboptimal solution) we used the hybrid algorithm, containing some elements of the best metaheuristics, based on the scatter search method.

3.2. The scatter-search method

The main ideas of this method are presented also in the work of James *et al.* (2005). It is based on a set evaluation so-called starting solutions. In the classical version the linear combination was used to determine indirect solutions among starting solutions. If the solution is represented as a permutation, using the linear permutation combination treated as a vector, it usually leads to a solution which is not a permutation. That's why in this work a procedure making a path linking two drawn solutions from a set containing starting elements was applied.

Algorithm 1. Scatter search (AScSr)

```
for $i$:=1 to number-iterationdo
    Step 1: Generate the set $S$ ($|S| = n$) of
            different random solutions;
    Step 2: For drawn $n/2$ pairs of starting
            solutions in $S$ apply
            the path-relinking procedure to
            generate the set
            $S'$ containing $n/2$ solutions
            situated on paths;
    Step 3: Apply an algorithm of local search
            in order to improve values
            of found solutions from the $S'$ set;
    Step 4: Add solutions from the set $S'$ to
            the set $S$. Leave in the set $S$
            at most $n$ solutions, removing
            from it the worst or the
            repeating solutions;
    Step 5: if $|S| < n$ then
            Draw a new solution to the $S$ set in
            such a way that elements
            from $S$ were different and $|S| = n$;
end {for}.
```

The basic element of the algorithm which affects directly the calculation time and values of determined solutions is made in Step 2 (*path-relinking procedure*).

3.2.1. *Path-relinking* procedure

In this procedure the elements' sequence between two fixed starting solutions is determined. Assuming one of them as current solution we fix a subset of the acceptable solutions set (called neighborhood)to find the closest solution which constitutes the first sequence element and the current solution. Next, the pattern is repeated as long as the second starting solution is obtained. This process should be managed in such a way that the path linking the selected solutions has a small number of elements. The procedure of determining and searching the neighborhood has a decisive influence here. Its elements are generated by "small" movements, which are made on the current solution.

In the presented algorithm the base of the procedure operation which generates the path linking two selected solutions is the scheme of multi-step fusion *MSXF*, an operator programmed in the work of Reeves and Yamada (1998). As a measure of distance between solutions (permutations) serving to manage the process of making the path between $\pi(i)$ and $\sigma(i)$ the Hamming's measure was adopted:

$$H(\pi,\sigma) = \text{number of } i \text{ such that } \pi(i) \neq \sigma(i).$$

The calculation complexity of determining the distance value is linear *O*(*n*), where *n* is the number of elements in a permutation. In the path-relinking procedure each next path element is determined by looking through the neighborhood, i.e. some subset of solutions space generated by insert moves.

3.2.2. Moves and neighborhood

Let $\pi = (\pi(1),\dots,\pi(n))$ be a *n* element permutation of the Φ_n set. The *insert move* i_l^k presents an element $\pi(k)$ (from the *k* position in π) on the position *l*, generating a permutation $i_l^k(\pi) = \pi_l^k$. If $l \geq k$, then:

$$\pi_l^k(i) = \begin{cases} \pi(i), & \text{if} \quad i < k \vee i > l, \\ \pi(i+1), & \text{if} \quad k \leq i < l, \\ \pi(k), & \text{if} \quad i = l, \end{cases} \tag{7}$$

and similarly in the case when $k > l$. We will call the insert move in brief *i-move*. Its calculation complexity equals $O(1)$. There are $n(n-1)$ of all such moves.

The permutation neighborhood $\pi \in \Phi_n$ generated by all *i-moves*:

$$\mathsf{N}(\pi) = \{i_l^k(\pi) : l,k = 1,2,\dots,n \text{ and } l \neq k\}. \tag{8}$$

The sequences of directly succeeding jobs:

$$\pi^B = (\pi(a),\pi(a+1),\dots,\pi(b-1),\pi(b)), \tag{9}$$

$(1 \leq a < b \leq n)$ in permutation $\pi \in \Phi_n$ we call subpermutations. A *weight* of a subpermutation $\mathsf{WT}(\pi^B) = \sum_{i=a}^{b} w_{\pi(i)} \cdot T_{\pi(i)}$, and a *length* $\mathsf{L}(\pi^B) = C_{\pi(b),m}$.

Let $\Phi_n(\pi^B) \subseteq \Phi_n$ be a set of permutations which can be generated from a permutation π by swapping an order of elements in a subpermutation π^B. The number of elements of the set $\Phi_n(\pi^B)$ equals $(b-a+1)!$.

Let F mean one of the defined above goal functions, i.e. WT (a weight) or L (a length). For a subpermutation π^B, let:

$$F^{mi}(\pi^B) \leq \min\{\mathsf{F}(\delta) : \delta \in \Phi_n(\pi^B)\}, \tag{10}$$

$$F^{mx}(\pi^B) \geq \max\{\mathsf{F}(\delta) : \delta \in \Phi_n(\pi^B)\}. \tag{11}$$

For any permutation $\gamma \in \Phi_n(\pi^B)$ a value of $F^{mi}(\pi^B)$ and $F^{mx}(\pi^B)$ are lower and upper bound of $\mathsf{F}(\gamma)$, therefore $\mathsf{F}(\gamma) \in [F^{mi}(\pi^B), F^{mx}(\pi^B)]$.

Subpermutations π^B we call $\theta - optimal$ $(0 < \theta \leq 1)$ due to the function F, if:

$$\mathsf{F}(\pi^B) \in [F^{mi}(\pi^B), F^{mi}(\pi^B) + \theta(F^{mx}(\pi^B) - F^{mi}(\pi^B))]. \tag{12}$$

Parameter θ will be determined experimentally.

3.3. Blocks of jobs in permutation

In any permutation we will determine two types of subpermutations called blocks in the further part of the paper:

1. T – blocks, consisting of on-time jobs;
2. D – blocks, consisting of late jobs.

On-time jobs blocks

Subpermutation $\pi^{\mathsf{T}} = (\pi(a),\pi(a+1),\dots,\pi(b))$ is a T *-block* in a permutation π, if:

(a) each job from π^{T} is on-time in the permutation π, so:

$$\forall j \in \{a,a+1,\dots,b\}, C_{\pi(j),m} \leq d_{\pi(j)},$$

(b) T -block is on-time, therefore the cost of its executing equals 0. Then, π^{T} is an optimal schedule due to the cost.

Late jobs blocks

Subpermutation of jobs $\pi^{\mathsf{D}} = (\pi(a),\pi(a+1),\dots,\pi(b))$ is a D *-block* in the permutation $\pi \in \Phi_n$, if:

(a') any job $\pi(j)$ $(j = a+1, a+2,\dots,b)$ moved into the first position in π^{D} (i.e. after executing an *insert* move i_a^j) is late,

(b') due to the cost a subpermutation π^{D} is θ*-optimal.*

From the constraint (**a'**) it follows that each job from a D -block, after moving into the first position in π^{D}, is late.

Algorithms of blocks determination in a permutation are described in the paper of Wodecki (2009). The number of blocks and the number of elements in each blocks depend on the parameter θ $(0 < \theta < 1)$.

Let $\mathsf{B} = \{\pi^1, \pi^2, \dots, \pi^t\}$ be a set of blocks (i.e. T and D blocks) in the permutation π and let $\mathsf{N}(\pi)$ be the neighborhood generated by insert moves (see (8)). We are applying restricted neighborhoods in algorithms. We are eliminating permutations generated from π by changing the order of elements in any block. Therefore, we take:

$$\mathsf{N}(\pi) = \mathsf{N}(\pi) \setminus \bigcup_{\delta \in \mathsf{B}} \Phi_n(\pi^\delta). \qquad (13)$$

Algorithm 2. The path-relinking generation procedure

```
$\pi_1, \pi_2$ – any solution of the $S$ set.
Let $x = q = \pi_1$; $T$ – temperature (parameter);
$\mathsf{N}(\pi)$ – solution neighborhood $\pi$ generated by all i-moves;
repeat
    For each $y_i \in \mathsf{N}(\pi)$ calculate the distance $H(y_i, \pi_2)$;
    Sort the element $y_i \in \mathsf{N}(\pi)$ ascendingly with reference to the measure $H(y_i, \pi_2)$;
        repeat
            Choose element $y_i$ of the set $\mathsf{N}(\pi)$ with the probability
            inversely proportional to the index $i$;
            Set $\mathsf{WT}(y_i)$;
            Accept $y_i$, if $\mathsf{WT}(y_i) \le \mathsf{WT}(x)$ or otherwise with the probability
            $P_T(y_i) = exp((\mathsf{WT}(x) - \mathsf{WT}(y_i)) / T)$;
            Change indexes $y_i$, from $i$ to $n$
            and indexes $y_k$, $k = i+1,...,n$ from $k$ to $k-1$;
        until $y_i$ is accepted;
        $x \leftarrow y_i$;
        if $\mathsf{WT}(x) < \mathsf{WT}(q)$ then $q \leftarrow x$;
    until Completion Condition;
    Starting from the $q$ solution makes the local optimization start
    the Descent search algorithm;
    return $q$; {q is the best element situated on the path}.
```

The algorithm of the relinking path generation stops its operation after obtaining the π_2 solution or after making in advance a fixed number of iterations (the *Completion Condition*).

The local optimization method

In order to improve the values, determined in **Algorithm 2**, of the best q element situated on the linking path the fast algorithm based on the descending method is applied.

Algorithm 3. **Descent search**

```
Let $y$ be the starting solution
$x_{best} \leftarrow y$;
repeat
    $x \leftarrow y$;
    find an element $y \in \mathsf{N}(x)$ such that
    $\mathsf{WT}(y) = min\{\mathsf{WT}(\pi) : \pi \in \mathsf{N}(x)\}$;
    if $\mathsf{WT}(y) < \mathsf{WT}(x_{best})$ then $x \leftarrow y$;
until $\mathsf{WT}(x) \ge \mathsf{WT}(y)$.
```

The conducted calculation experiments proved that in **Algorithm 3** not more than n (the number of permutation elements) iterations is made.

4. Computational experiments

To conduct the statistical analysis of the algorithm operation this method was also tested on test data. The results were compared with the NEH (Navaz *et al.* 1983) algorithm. In Taillard's work (Taillard 1993) there is presented the commonly used these days method of data generation to evaluate approximation algorithms used to solve different combinatorial optimization problems. In its primary version, it enables to find instances for the flow shop problem only on the based of the criterion $C_{\max}$. While applying this method the times to conduct the operation were generated. An integer weight was generated from the uniform distribution [1,10]. The required completion times were generated using the following procedure:

Step 1: Calculate:

$$P_i = \sum_{j=1}^{m} p_{i,j}, \; i = 1,2,\ldots,n,$$

the total time of carrying out the task $i \in \mathsf{J}$ on machines.

Step 2: For each task $i \in \mathsf{J}$ determine the required completion time:

$$d_i = P_i \cdot (1 + \mu \cdot 3).$$

The parameter μ is the random variable realization of a uniform schedule on the [0,1] distance. This generator is described in the (Taillard 1993). A similar way of data determination is used in Hasija and Rejendra (2004) too. The generated data form 8 groups win a size $n \times m$: 20x5, 20x10, 50x5, 50x10, 50x20, 100x5, 100x10 and 100x20. For each size 25 examples were determined, so all data embraced 200 examples.

The scatter search AScSr algorithm was programmed in C++ language. The calculation experiments were carried out on the IBM microcomputer with a 1.8 GHz processor. To determine the starting solution the NEH algorithm was employed. In literature, there are neither data nor results to compare with this problem. Therefore, an additional series of instances in a small size

$n \times m$ was generated: 8x4, 8x6, 10x4, 10x6, 12x4, 12x6 and 12x8 – 10 examples for each size. The obtained results for the data of the NEH algorithm were compared with optimal values determined by a total search. The average relative error equals 8.62%. Before starting the basic test to determine the AScSr algorithm's parameters calculations were made on a small data group – one example from each size. On this basis it was assumed that the parameter θ (in the block definition) would be equal to 0.2.

The basic calculation tests were carried out for two versions of an algorithm: with blocks AScSr+B (neighborhood (13)) and without blocks AScSr (neighborhood (8)). Each algorithm made 2 000 iterations, parameter $T = 60$. The obtained results were compared with the NEH algorithms solutions. The average relative errors (δ_{aprd}) as well as the average calculation times (t_{aprd}) are presented in Table 5.

Due to the obtained results it can be stated that the blocks application brought about a considerable improvement. The algorithm with blocks determines better solutions in a shorter time. The average improvement of the starting solution (determined by NEH) equals 6.20%; whereas for the algorithm without blocks it equals 4.55%. Furthermore, the average time of this algorithm operation is about 22% smaller. The average calculation times of one instance equals 7.25 and 9.13 seconds, respectively. The time calculation shortening results from the fact that in case of the algorithm without blocks the whole neighborhood is searched. The number of its elements is $O(n^2)$ and the calculation time of the goal function value for one element is $O(n \cdot m)$. In turn, the procedure of blocks determination in a permutation has a calculation complexity $O(n^2 m)$. Besides, the calculation experiments proved that the restricted neighborhood which is generated with the use of blocks elimination properties is about 35% smaller than the whole neighborhood. Therefore, elimination, through filtration (heuristic indirect search) of many non-perspective elements causes values improvement of fixed solutions and the calculation time shortening.

Table 5. Average relative errors (δ_{aprd}) and average calculation times (t_{aprd})

Problem size	Algorithm AScSr+B		Algorithm AScSr	
	δ_{aprd}	t_{aprd}	δ_{aprd}	t_{aprd}
20x5	–3.48	0.9	–3.24	0.8
20x10	–3.82	1.7	–3.87	2.1
50x5	–5.28	2.1	–4.12	2.2
50x10	–5.71	3.1	–5.83	3.9
50x20	–6.12	3.8	–5.34	4.6
100x5	–8.31	8.4	–8.29	9.3
100x10	–8.76	13.7	–7.64	18.6
100x20	–8.12	24.3	–8.76	31.5
average	**–6.20**	**7.25**	**–4.55**	**9.13**

5. Computational results for Kunice case study

For a Microsoft Project schedule the value of the cost function was 4668 Euro. Works delays and costs are given in Tables 6 and 7. Halls was developed in the sequence (NAWA1, NAWA2, NAWA3). Six works are delayed.

However, the optimal value (0 euro) was obtained for the schedule (NAWA2, NAWA1, NAWA3) by both proposed AScSr and AScSr+B scatter search algorithms. No works are delayed. The value of the goal function must not be negative, so we have the certainty that the zero value is the optimal one.

6. Conclusions

This paper provides a practical scatter search algorithm for the total weighted tardiness flow shop problem, as a mathematical model of construction project management. We propose new properties of blocks of jobs, which makes possible to construct sub-neighborhoods of a smaller size than classical ones in the local search algorithms. Computational experiments show that the proposed methodology with blocks provides solutions which are much better than obtained by the classic tabu search approach without blocks.

Table 6. Stages delay times (in days)

Hall	Stages				
	Ground works	Fundamental works	Steel construction montage	Roof casing	Walls casing
NAWA1	0	0	0	0	0
NAWA2	0	0	0	0	1
NAWA3	0	0	0	2	2

Table 7. Stages delay costs (in euro)

Hall	Stages				
	Ground works	Fundamental works	Steel construction montage	Roof casing	Walls casing
NAWA1	0	0	0	0	0
NAWA2	0	0	0	0	756
NAWA3	0	0	0	384	3528

References

Adeli, H.; Karim, A. 1997. Scheduling cost optimisation and neural dynamics model for construction, *Journal of Construction Engineering and Management* ASCE 123(4): 450–458. http://dx.doi.org/10.1061/(ASCE)0733-9364(1997)123:4(450)

Bulfin, R. L.; M'Hallah, R. 2003. Minimizing the weighted number of tardy jobs on two-machineflow shop, *Computers & Operations Research* 30(12): 1887–1900. http://dx.doi.org/10.1016/S0305-0548(02)00114-4

Burduk, A.; Chlebus, E. 2009. Methods of risk evaluation in manufacturing systems, *Archives of Civil and Mechanical Engineering* 9(3): 17–30. http://dx.doi.org/10.1016/S1644-9665(12)60215-5

Chen, S.-M.; Griffis, F. H.; Chen, P.-H.; Chang, L.-M. 2012. Simulation and analytical techniques for construction resource planning and scheduling, *Automation in Construction* 21: 99–113. http://dx.doi.org/10.1016/j.autcon.2011.05.018

Chung, C.-S.; Flynn, J.; Kirca, Ö. 2006. A branch and bound algorithm to minimize the total tardiness for ***m***-machine permutation flowshop problems, *European Journal of Operational Research* 174(1): 1–10. http://dx.doi.org/10.1016/j.ejor.2004.12.023

El-Rayes, K.; Moselhi, O. 2001. Optimum resource utilization for repetitive construction projects, *Journal of Construction Engineering and Management* ASCE 127(1): 18–27. http://dx.doi.org/10.1061/(ASCE)0733-9364(2001)127:1(18)

Garey, M. R.; Johnson, D. S.; Sethi, R. 1976. The complexity of flowshop and jobshop scheduling, *Mathematics of Operations Research* 1(2): 117–129. http://dx.doi.org/10.1287/moor.1.2.117

Glover, F.; Laguna, M. 1997. *Tabu Search*. Kluwer Academic Publishers. 408 p. http://dx.doi.org/10.1007/978-1-4615-6089-0

Grabowski, J.; Wodecki, M. 2004. A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion, *Computers & Operations Research* 31(11): 1891–1909. http://dx.doi.org/10.1016/S0305-0548(03)00145-X

Hariri, A. M. A.; Potts, C. N. 1989. A branch and bound algorithm to minimize the number of late jobs in a permutation flow-shop, *European Journal of Operational Research* 38(2): 227. http://dx.doi.org/10.1016/0377-2217(89)90107-0

Harris, R. B.; Ioannou, P. G. 1998. Scheduling projects with repeating activities, *Journal of Construction Engineering and Management* ASCE 124(4): 269–278. http://dx.doi.org/10.1061/(ASCE)0733-9364(1998)124:4(269)

Hasija, S.; Rajendran, C. 2004. Scheduling in flowshops to minimize total tardiness of jobs, *International Journal of Production Research* 42(11): 2289–2301. http://dx.doi.org/10.1080/00207540310001657595

Ishibuchi, H.; Misaki, S.; Tanaka, H. 1995. Modified simulated annealing algorithms for the flow shop sequencing problem, *European Journal of Operational Research* 81(2): 388–398. http://dx.doi.org/10.1016/0377-2217(93)E0235-P

James, T.; Rego, C.; Glover, F. 2005. Sequential and parallel path-relinking algorithms for the quadratic assignment problem, *IEEE Intelligent Systems* 20(4): 58–65. http://dx.doi.org/10.1109/MIS.2005.74

Jiang, A.; Issa, R. R. A.; Malek, M. 2011. Construction project cash flow planning using the Pareto optimality efficiency network model, *Journal of Civil Engineering and Management* 17(4): 510–519. http://dx.doi.org/10.3846/13923730.2011.604537

Johnson, S. M. 1954. Optimal two- and three-stage production schedules with setup times included, *Naval Research Logistics Querterly* 1(1): 61–68. http://dx.doi.org/10.1002/nav.3800010110

Lageweg, B. J.; Lenstra, J. K.; Rinnooy Kann, A. H. G. 1978. A general bouding scheme for the permutation flow-shop-problems, *Operations Research* 26(1): 53–67. http://dx.doi.org/10.1287/opre.26.1.53

Lenstra, J. K.; Rinnoy Kann, A. H. G.; Brucker, P. 1977. Complexity of machine scheduling problems, *Annals of Discrete Mathematics* 1: 343–362. http://dx.doi.org/10.1016/S0167-5060(08)70743-X

Liu, S.-S.; Wang, C.-J. 2011. Optimizing project selection and scheduling problems with time-dependent resource constraints, *Automation in Construction* 20(8): 1110–1119. http://dx.doi.org/10.1016/j.autcon.2011.04.012

Liu, S.-S.; Wang, C.-J. 2012. Optimizing linear project scheduling with multi-skilled crews, *Automation in Construction* 24: 16–23. http://dx.doi.org/10.1016/j.autcon.2011.12.009

Medineckiene, M.; Zavadskas, E. K.; Turskis, Z. 2011. Dwelling selection by applying fuzzy game theory. *Archives of Civil and Mechanical Engineering* 11(3): 681–697. http://dx.doi.org/10.1016/S1644-9665(12)60109-5

Nawaz, M.; Enscore, E. E.; Ham, I. 1983. A heuristic algorithm for the *m*-machine, *n*-job flow-shop sequencing problem, *OMEGA* 11(1): 91–95. http://dx.doi.org/10.1016/0305-0483(83)90088-9

Nowakowski, T. 2010. Problems with analyzing operational data uncertainty, *Archives of Civil and Mechanical Engineering* 10(3): 95–109. http://dx.doi.org/10.1016/S1644-9665(12)60139-3

Onwubolu, G.; Davendra, D. 2006. Scheduling flow shop using differential evolution algorithm, *European Journal of Operational Research* 171(2): 674–692. http://dx.doi.org/10.1016/j.ejor.2004.08.043

Reeves, C. R.; Yamada, T. 1998. Solving the C_sum permutation flowshop scheduling problem by genetic local search, in *The 1998 IEEE International Conference on Evolutionary Computation*, Piscataway, NJ, USA, 230–234.

Russell, A. D.; Caselton, W. F. 1988. Extensions to linear scheduling optimisation, *Journal of Construction Engineering and Management* ASCE 114(1): 36–52. http://dx.doi.org/10.1061/(ASCE)0733-9364(1988)114:1 (36)

Russell, A. D.; Wong, W. C. M. 1993. New generation of planning structures, *Journal of Construction Engineering and Management* ASCE 119(2): 196–214. http://dx.doi.org/10.1061/(ASCE)0733-9364(1993)119:2(196)

Sarker, B. R.; Egbelu, P. J.; Liao, T. W.; Yu, J. 2012. Planning and design models for construction industry: A critical survey, *Automation in Construction* 22: 123–134. http://dx.doi.org/10.1016/j.autcon.2011.09.011

Schaller, J. 2005. Note on minimizing total tardiness in a two-machine flowshop, *Computers & Operations Research* 32(12): 3273–3281. http://dx.doi.org/10.1016/j.cor.2004.05.012

Senouci, A. B.; Eldin N. N. 1996. Dynamic programming approach to scheduling of nonserial linear project, *Journal of Computing in Civil Engineering* ASCE 10(2): 106–114. http://dx.doi.org/10.1061/(ASCE)0887-3801(1996)10:2(106)

Taillard, E. 1993. Benchmarks for basic scheduling problems, *European Journal of Operational Research* 64(2): 278–285. http://dx.doi.org/10.1016/0377-2217(93)90182-M

Ucar, H.; Tasgetiren, M. F. 2006. A particle swarm optimization algorithm for permutation flow shop sequencing problem with the number of tardy jobs criterion, in *Proc. of the 5th International Symposium on Intelligent Manufacturing Systems IMS'2006*, Sakarya, Turkey, 1100–1120.

Vallada, E.; Ruiz, R.; Minella, G. 2008. Minimizing total tardiness in the *m*-machine flowshop problem: A review and evaluotion of heuristics and metaheuristics, *Computers & Operations Research* 35(4): 1350–1373. http://dx.doi.org/10.1016/j.cor.2006.08.016

Wodecki, M. 2009. A block approach to earliness-tardiness scheduling problems, *The International Journal of Advanced Manufacturing Technology* 40(7–8): 797–807. http://dx.doi.org/10.1007/s00170-008-1395-7

Zavadskas, E. K.; Turskis, Z.; Vilutiene, T. 2010a. Multiple criteria analysis of foundation instalment alternatives by applying Additive Ratio Assessment (ARAS) method, *Archives of Civil and Mechanical Engineering* 10(3): 123–141. http://dx.doi.org/10.1016/S1644-9665(12)60141-1

Zavadskas, E. K.; Turskis, Z.; Tamošaitiene, J. 2010b. Risk assessment of construction projects, *Journal of Civil Engineering and Management* 16(1): 33–46. http://dx.doi.org/10.3846/jcem.2010.03

Zavadskas, E. K.; Turskis, Z.; Tamosaitiene, J. 2011. Selection of construction enterprises management strategy based on the SWOT and multi-criteria analysis, *Archives of Civil and Mechanical Engineering* 11(4): 1063–1082. http://dx.doi.org/10.1016/S1644-9665(12)60096-X

Zhang, X. Q.; Gao, H. 2010. Optimal performance-based building facility management, *Computer-Aided Civil and Infrastructure Engineering* 25(4): 269–284. http://dx.doi.org/10.1111/j.1467-8667.2009.00633.x

Wojciech BOŻEJKO. Dr Habil at Wroclaw University of Technology. He obtained PhD in Wroclaw University of Technology, Institute of Computer Engineering, Control and Robotics in 2003 and D.Sc. (habilitation) in Wrocław University of Technology, Faculty of Electronics, in 2011. Author of over 130 papers in journals and conference proceedings from the field of parallel processing, scheduling and optimization. He is interested in parallel algorithms, discrete optimization, scheduling and supercomputing. He is also a qualified musician, graduated (M.A.) from the Academy of Music in Wrocław, Poland, in a specialization of piano.

Zdzisław HEJDUCKI. Prof, Dr Habil at Institute of Building Engineering of Wroclaw University of Technology, Poland. Research interests: civil engineering and management, special scheduling methods and the optimization of discrete processes occurring directly in construction, focusing mainly on the application of artificial intelligence methods (simulating annealing, genetic algorithm, tabu search, neural networks).

Mieczysław WODECKI. Dr Habil at Institute of Computer Science University of Wrocław. He obtained PhD in Wroclaw University of Technology, Institute of Computer Engineering, Control and Robotics in 1988 and D.Sc. (habilitation) in Wrocław University of Technology, Faculty of Electronics, in 2011. He is an author of over 140 papers (and over 45 citations on ISI Web of Knowledge) in journals and conference proceedings from the field of parallel processing, scheduling and optimization. He is interested in discrete optimization and parallel algorithms.