Management and Production Engineering Review
Volume 2 • Number 1 • March 2011 • pp. 3–9

mper

PROJECT MANAGEMENT IN BUILDING PROCESS WITH UNCERTAIN TASKS TIMES

Wojciech Bożejko[1], Zdzisław Hejducki[1], Paweł Rajba[2], Mieczysław Wodecki[2]

[1] *Wrocław University of Technology*
[2] *University of Wrocław*

Corresponding author:
Mieczysław Wodecki
University of Wrocław
Institute of Computer Science
Joliot-Curie 15, 50-383 Wrocław, Poland
phone: +48 71 375 78 00
e-mail: mwd@ii.uni.wroc.pl

Received: 5 February 2011
Accepted: 7 March 2011

ABSTRACT
In this paper there is presented a problem of scheduling of construction work in which certain projects must be executed. Every work consists of projects executed by separate teams. In a linear system the sequence of works is the same for every project. Uncertain tasks times are represented by fuzzy numbers or distribution of random variables. We present a tabu search algorithm and computational experiments which are aimed at checking the sustainability of set solutions.

KEYWORDS
planning of building, linear system, uncertain tasks times, tabu search algorithm.

Introduction

An optimal planning of bulding projects is an important task as it influences the effectivenes of executive companies in a crucial way. In a negotiation process of construction contracts there appears necessity to define the dates of project completion. It is a difficult poblem as the level of uncertainty concerning different, constantly changing parameters is rather high. Not meeting the deadlines results in generating losses (contractual penalties or unused resources). Thus, there appears the need for modelling of construction tasks that would be the most precise projection of the course of bulding processes [1, 2]. It leads to a complex descrete – continuous optimization problems with uncertain parameteres and irregular goal functions. While converting the issues of construction processes into a field of classical theory concerning sequencing of tasks one may encounter many problems connected to choosing an appropriate model and adequate algorithm. They are most commonly completely new, *strongly NP-hard* problems of combinatorial optimization.

An integrated part of many management systems is planning of bulding processes in a linear system [3]. It concerns realization of complex projects consiting of many identical tasks undertaken by specialised teams. It is an equivalent of flow production in industry. Projects are represented by tasks, teams by machies and works executed by teams are represented by operations. The order of works executions at the objects are represented by a technological order.

Taking into consideration the implementation of new techniques and technologies, uniquness, atmospheric and geological conditions, it is often impossible to define the value of certain parameters explicitely. In such cases we have to deal with taking decisions at a high level of uncertainty. Uncertainty of data influences directly the degree of the risk. Moreover, during the process of project realization it might appear that some parameters differ from provisionally accepted (typical) and with the lack of sustainability of solution it leads to solutions which are completely useless in practice. Failures resulting from direct implementation of classic deterministic algorithms indicate the necessity of consideration of

uncertainty in the beginning of the whole process of model building and during the construction of an algorithm itself.

Poblems of taking decisions under uncertainty are solved by application of probabilistic method or through fuzzy sets theory. In the first case [4, 5] knowledge of distribution of random variables is of crucial importance. Some pocesses are characterised with randomity by nature. They depend on weather conditions, traffic intensity, number of accidents, geological conditions, device's failure, etc. If they, nevertheless, posses certain "history", it is possible to define their distribution on the basis of statistical data.

In many issues the uncertainty of data is not of random nature but it results from uniqueness of a process, errorr in measurement, etc. In such a case a natural method of representing uncertainty are fuzzy numbers [6, 7]. In this case a huge problem is posed by a proper choice of membership function and deffuzification method. They have crucial influence on the quality of taken solutions.

In this paper we consider the problem of scheduling of bulding tasks realized in a linear system. They consitute an important part of construction practice issue, having a marked and direct impact on the final costs of project realization. We propose algorithms for solving certain problems based on a tabu searched method and modification of this algorithm for a case when the deadlines are uncertain. We compare the sustainability of solutions in case where uncertain data are represented by random variables with a normal distibution or fuzzy numbers in a threetuple repesentation.

Linear system in bulding

We examine a building project (**BP** in short) consisting in execution of n objects from a set

$$O = \{O^1, O^2 \ldots, O^n\},$$

by m teams from a set

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

Every object $O^i \in O$ is a sequence of m projects

$$O^i = [P_1^i, P_2^i, \ldots, P_m^i],$$

where project P_j^i $(i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, m)$ is executed by B_j team in p_j^i time. Projects in $O^i \in O$ object should be performed in a fixed technological order, i.e. any P_j^i project is to be executed after the completion of P_{j-1}^i, and before the beginning of P_{j+1}^i $(2 \leq j \leq m-1)$. The following constraints must be satisfied:

(i) every project (at the object) can be executed only by one, defined by a technological sequence, team,

(ii) none of the teams can execute more than one work at a time,

(iii) technological order must be maintained at every object,

(iv) execution of any work cannot be terminated before its completion.

Let π be a certain pemutation of objects (elements of O set). The permutation will be some permutation of objects (elements of set). This permutation defines the sequence of execution of particular works at objects i.e. $B_j \in B$ team executes $P_j^{\pi(i)}$ projects at $\pi(i) \in O$ object, only after execution of $P_j^{\pi(1)}, P_j^{\pi(1)}, \ldots, P_j^{\pi(i-1)}$ projects successively at $\pi(1), \ldots, \pi(i-1)$ objects, but before execution of $P_j^{\pi(i+1)}, P_j^{\pi(i+2)}, \ldots, P_j^{\pi(n)}$ works at $\pi(i+1), \ldots, \pi(n)$ objects. Let us denote by Φ a set of all possible permutations of objects. The cardinality of this set equals $n!$.

If the works at objects are executed in $\pi \in \Phi$ order and $p_{\pi(i),j}$ is the time of execution of $P_j^{\pi(i)}$ $(i \in O,\ j \in B)$ project, then the moment of completion of this work $C_{\pi(i),j}$ can be determined from the following recurring dependency:

$$C_{\pi(i),j} = \begin{cases} \sum_{k=1}^{i} p_{\pi(i),j}, & j = 1, \\ C_{\pi(i),j-1} + p_{\pi(i),j}, & i = 1, j > 1, \\ \max\{C_{\pi(i),j-1}, C_{\pi(i-1),j}\} + p_{\pi(i),j}, & i > 1, j > 1, \end{cases} \quad (1)$$

and the moment of the beginning of its execution

$$S_{\pi(i),j} = C_{\pi(i),j} - p_{\pi(i),j}. \quad (2)$$

One can easily check that defined by (1) and (2) moments of beginning and completion of projects at objects fulfil the constraints (i)–(iv), thus they are acceptable solutions of **BP** problem.

Model of the above described project is known in a scheduling theory as a *flow shop problem*. If we examine the criterion of minimization of time completion of all the objects ($C_{\max}$), then this problem can be qualified as *NP-hard* [8]. It is usually solved by heuristic methods. The use of simulated annealing is presented, e.g., in Osman and Potts [9], Ogbu and Smith [10], Bożejko and Wodecki [11] (parallel algorithm), tabu search in Nowicki and Smutnicki [12], Grabowski and Wodecki [13], genetic algorithm in Reeves [14].

A very important criterion in a building process is meeting the deadline or a possible minimization of penalties for breach of contractual terms.

For P_j^i project let d_{ij}, w_{ij} be adequately: a deadline time for execution of works set in contract time and a coefficient of penalty for delays. Three $\theta = \langle p, d, w \rangle$, where: $p = [p_{i,j}]_{n\times m}$ – matrix of projects execution time, $d = [d_{i,j}]_{n\times m}$ – terms matrix, $w = [w_{i,j}]_{n\times m}$ – matrix of penalty coefficient, is an instance of deterministic data.

If $\pi \in \Phi$ is a sequence of objects execution and $C_{\pi(i),j}$ is a term of project completion $P_j^{\pi(i)}$, then

$$U_{\pi(i),j} = \begin{cases} 1, & \text{if } C_{\pi(i),j} > d_{\pi(i),j} \\ 0, & \text{if } C_{\pi(i),j} \le d_{\pi(i),j} \end{cases} \quad (3)$$

is a delay, and $w_{\pi(i),j} \cdot U_{\pi(i),j}$ is a penalty for delay. Then

$$W(\pi) = \sum_{i=1}^{n}\sum_{i=1}^{m} w_{\pi(i),j} \cdot U_{\pi(i),j} \quad (4)$$

is a sum of penalties for not meeting the deadline of project completion (in short, cost of permutation π).

The problem examined in this work consists in determining optimal sequence of execution of objects in a linear system minimizing function (4). It resolves to determining permutation $\pi^* \in \Phi$ that

$$W(\pi^*) = \min\{W(\pi)\colon \pi \in \Phi\}.$$

The problem will be defined in short as **PFS**. For it is an equivalent of *NP-hard* one machine problem of sequencing tasks $1||\sum w_i U_i$ [15]. Currently there are not known optimal algorithms for solving problems of multinominal computational complexity. This is why for finding a solution of **PFS** problem we use a heuristic tabu search algorithm.

Classic tabu search algorithm

In solving *NP-hard* problems of discrete optimization we almost always use approximate algorithms. The solutions given by these algorithms are, in their appliance, fully satisfying (they often differ from the best known solutions by less then 1%). Most of them belong to the local search methods group. Their acting consists in viewing in sequence a subset of a set of acceptable solutions, and in pointing out the best one according to a determined criterion. One of this method realizations is the tabu search, whose basic criterions are:

- *neighborhood* – a subset of a set of acceptable solutions, whose elements are rigorously analyzed,
- *move* – a function that converts one solution into another one,
- *tabu list* – a list containing the attributes of a certain number of solutions analyzed recently,
- *ending condition* – most of the time fixed by the number of algorithm iterations.

Let $\pi \in \Phi$ be any (starting) permutation, L_{TS} a tabu list, W costs function, and π^* the best solution found at this moment (the starting solution and π^* can be any permutation).

Algorithm Tabu Search (TS)

repeat
Determinate the neighborhood $N(\pi)$ of the permutation π;
Remove from $N(\pi)$ the permutations forbidden by the L_{TS} list;
Determinate the permutation $\delta \in N(\pi)$, in which
$W(\delta) = \min\{W(\beta)\colon \beta \in N(\pi)\}$;
if $(W(\delta) < W(\pi^*))$ ***then*** $\pi^* := \delta$;
Include δ parameters on the L_{TS} list;
$\pi := \delta$
until (*ending condition*).

The computational complexity of the algorithm depends mostly on the way the neighborhood is generated and viewed. Below we present in details the basics elements of the algorithm.

The move and the neighborhood

Let $\pi = (\pi(1), \ldots, \pi(n))$ be any permutation from the Φ, and

$$L(\pi) = \{\pi(i)\colon C_{\pi(i),m} > d_{\pi(i),m}\},$$

a *set of late tasks* in π.

By π_l^k $(l = 1, 2, \ldots, k-1, k+1, \ldots, n)$ we mark a permutation received from π by changing in π the element $\pi(k)$ and $\pi(l)$. We can say at that point that the permutation π_l^k was generated from π by a *swap move* (*s-move*) s_l^k (it means that the permutation $\pi_l^k = s_l^k(\pi)$). Then, let $M(\pi(k))$ be a set of all the s-moves of the $\pi(k)$ element. By

$$M(\pi) = \bigcup_{\pi(k)\in L(\pi)} M(\pi(k)),$$

we mean an *s-moves* set of the late elements π in the pemutation. The power of the set $M(\pi)$ is top-bounded by $n(n-1)/2$.

The neighborhood $\pi \in \Phi$ is the permutation set

$$N(\pi) = \{s_l^k(\pi)\colon s_l^k \in M(\pi)\}.$$

While implementing the algorithm, we remove from the neighborhood the permutations whose attributes are on the forbidden attributes list L_{TS}.

The tabu list

In order to avoid generating a cycle (by returning to the same permutation after a small number of algorithm iterations), some attributes of every move are saved on a tabu list. It is operated according to the FIFO queue. By making the $s_j^r \in M(\pi)$ (generating from $\pi \in \Phi$ the permutation π_j^r) we write on the tabu list L_{TS} of this move's attributes, the tuple $(\pi(r), j, W(\pi_j^r))$.

Suppose, that we analyze the move $s_l^k \in M(\beta)$ generating from $\beta \in \Phi$ the β_l^k permutation. If the tuple (r, j, Ψ), such that $\beta(k) = r$, $l = j$ and $W(\beta_l^k) \geq \Psi$ is on the L_{TS} list, such a move is forbidden and removed from the $M(\beta)$ set. The only parameter of this list is its length, the number of the elements it contains. There are many realizations of the tabu list in the bibliography.

Uncertain tasks times

We assume that times of tasks execution are not deterministic. They will be represented with the use of fuzzy numbes or random vaiable.

Fuzzy tasks times

In this paper the fuzzy tasks times are represented by a triangular membership function μ (i.e. 3-tuple $\widehat{p}_{i,j} = (p_{i,j}^{\min}, p_{i,j}^{\text{med}}, p_{i,j}^{\max})$ $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$ (see Fig. 1) with the following properties:

- $(p_{i,j}^{\min} \leq p_{i,j}^{\text{med}} \leq p_{i,j}^{\max})$,
- $\mu(a) = 0$ for $a \leq p_{i,j}^{\min}$ or $a \geq p_{i,j}^{\max}$,
- $\mu(p_{i,j}^{\text{med}}) = 1$,
- μ is increasing on $[p_{i,j}^{\min}, p_{i,j}^{\text{med}}]$ and decreasing on $[p_{i,j}^{\text{med}}, p_{i,j}^{\max}]$.

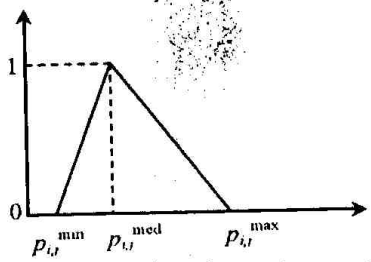

Fig. 1. A triangular $(p_{i,j}^{\min}, p_{i,j}^{\text{med}}, p_{i,j}^{\max})$ fuzzy membership function.

The addition of fuzzy numbers

$$\widehat{a} = (a_1, a_2, a_3) \quad \text{and} \quad \widehat{b} = (b_1, b_2, b_3),$$

can be derived from the extension principle and it is as follows [16]

$$\widehat{a} + \widehat{b} = (a_1 + b_1, a_2 + b_2, a_3 + b_3).$$

Similarly

$$\max\{\widehat{a}, \widehat{b}\} = (\max\{a_1, b_1\}, \max\{a_2, b_2\}, \max\{a_3, b_3\}).$$

Let $\theta = \langle p, d, w \rangle$ be an example of deterministic data for **PFS** problem. We assume that times of work P_j^i, $i \in O$, $j \in B$ execution are fuzzy numbers

$$\widehat{p}_{i,j} = (p_{i,j}^{\min}, p_{i,j}^{\text{med}}, p_{i,j}^{\max}),$$

where $p_{i,j}^{\min} = p_{i,j} - \lfloor p_{i,j}/6 \rfloor$, $p_{i,j}^{\text{med}} = p_{i,j}$ and

$$p_{i,j}^{\max} = p_{i,j} + \lfloor p_{i,j}/3 \rfloor.$$

The three $\bar{\theta} = \langle \widehat{p}, d, w \rangle$, where $\widehat{p} = [\widehat{p}_{i,j}]_{n \times m}$ is a matrix of fuzzy numbers, are called fuzzy data, and the problem – fuzzy one (**PFSF** in short).

If permutation $\pi \in \Phi$ and the time of the execution is determined by a fuzzy number

$$\widehat{p}_{\pi(i),j} = (p_{\pi(i),j}^{\min}, p_{\pi(i),j}^{\text{med}}, p_{\pi(i),j}^{\max}),$$

then its finishing time is a fuzzy number in the form of:

$$\widehat{C}_{\pi(i),j} = (\widehat{C}_{\pi(i),j}^{\min}, \widehat{C}_{\pi(i),j}^{\text{med}}, \widehat{C}_{\pi(i),j}^{\max}),$$

where $C_{\pi(i),j}^{\min}$, $C_{\pi(i),j}^{\text{med}}$ and $C_{\pi(i),j}^{\text{med}}$ can be determined from the following recurrent formulas:

$$C_{\pi(i),j}^{\min} = \max\{C_{\pi(i-1),j}^{\min}, C_{\pi(i),j-1}^{\min}\} + p_{\pi(i),j}^{\min},$$

$$C_{\pi(i),j}^{\text{med}} = \max\{C_{\pi(i-1),j}^{\text{med}}, C_{\pi(i),j-1}^{\text{med}}\} + p_{\pi(i),j}^{\text{med}},$$

$$C_{\pi(i),j}^{\max} = \max\{C_{\pi(i-1),j}^{\max}, C_{\pi(i),j-1}^{\max}\} + p_{\pi(i),j}^{\max},$$

with the initial conditions

$$C_{\pi(0),j}^{\min} = C_{\pi(0),j}^{\text{med}} = C_{\pi(0),j}^{\max}, \quad j = 1, 2, \ldots, m,$$

$$C_{\pi(i),0}^{\min} = C_{\pi(i),0}^{\text{med}} = C_{\pi(i),o}^{\max}, \quad i = 1, 2, \ldots, n.$$

We perform a deffuzification

$$t_{\pi(i),j} = \frac{1}{4}\left(C_{\pi(i),j}^{\min} + C_{\pi(i),j}^{\text{med}} + C_{\pi(i),j}^{\text{med}} + C_{\pi(i),j}^{\max}\right). \quad (5)$$

The equivalent of delay (3) is

$$v_{\pi(i),j} = \begin{cases} 1, & \text{if } t_{\pi(i),j} > d_{\pi(i),j}, \\ 0, & \text{if } t_{\pi(i),j} \leq d_{\pi(i),j}. \end{cases}$$

In case of fuzzy data, the equivalent (4) is function

$$WF(\pi) = \sum_{j=1}^{m} \sum_{j=1}^{n} w_{\pi(i),j} v_{\pi(i),j}. \quad (6)$$

An algorithm of solving **PFSF** problem (with a goal function (6)) is called fuzzy, **TSF** in short.

Probabilistics tasks times

Let $\theta = \langle p, d, w\rangle$ be an example of deterministic data for **PFS** problem. We assume that times of execution of works P_j^i, $i \in O$, $j \in B$ are independent random variables with a normal distribution, i.e. $\tilde{p}_{i,j} \sim N(p_{i,j}, \sigma_{i,j})$. The expected value of times $E(\tilde{p}_{i,j}) = p_{i,j}$. Then data $\tilde{\theta} = \langle \tilde{p}, d, w\rangle$, where $\tilde{p} = [\tilde{p}_{i,j}]_{n\times m}$ is a matrix of random variables, we call a probabilistic data, and the problem – probabilistic (**PFSR** in short).

Let $\pi \in \Phi$ be some sequence of tasks execution at objects. In order to simplify the calculations we assume that moments of completion of separate works have also a normal distribution

$$\tilde{C}_{\pi(i),j} \sim N\left(C_{\pi(i),j}, \alpha\sqrt{C^2_{\pi(i),j}}\right),$$

where

$$C^2_{\pi(i),j} = \begin{cases} \sum\limits_{k=1}^{i} p^2_{\pi(k),j}, & j = 1, \\ C^2_{\pi(i),j-1} + p^2_{\pi(i),j}, & i = 1, j > 1, \\ \max\{C^2_{\pi(i),j-1}, C^2_{\pi(i-1),j}\} + p^2_{\pi(i),j}, & i > 1, j > 1, \end{cases}$$

and α parameter is experimentally determined. The equivalent of delays (3) are random variable

$$\tilde{U}_{\pi(i),j} = \begin{cases} 1, & \text{if } \tilde{C}_{\pi(i),j} > d_{\pi(i),j}, \\ 0, & \text{if } \tilde{C}_{\pi(i),j} \le d_{\pi(i),j}. \end{cases}$$

By solving a **PFSR** problem (with a random times of tasks execution) for a cost function (4) we assume

$$WP(\pi) = \sum_{j=1}^{m}\sum_{i=1}^{n} w_{\pi(i),j}(E(\tilde{U}_{\pi(i),j}) + D^2(\tilde{U}_{\pi(i),j})),$$

where $E(\tilde{U}_{\pi(i),j})$ is an expected value, $D^2(\tilde{U}_{\pi(i),j})$, a variation of random variable $\tilde{U}_{\pi(i),j}, i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$. Because

$$E(\tilde{U}_{\pi(i),j}) = P(\tilde{C}_{\pi(i),j} > d_{\pi(i),j} = 1 - F_{\tilde{C}_{\pi(i),j}}(d_{\pi(i),j}),$$

$$D^2(\tilde{U}_{\pi(i),j}) = F_{\tilde{C}_{\pi(i),j}}(d_{\pi(i),j})(1 - F_{\tilde{C}_{\pi(i),j}}(d_{\pi(i),j})),$$

where F_X is distribuant of random variable X with a normal distribution. Finally

$$WF(\pi) = \sum_{j=1}^{m}\sum_{i=1}^{n} w_{\pi(i),j}\left(1 - (F_{\tilde{C}_{\pi(i),j}}(d_{\pi(i),j}))^2\right). \quad (7)$$

Tabu search algorithm of solving rozwiązywania problemu **PFSR** problem (with a goal function (7)) we call a probabilistic one, **TSR** in short.

Sustainability of algorithms

Sustainability is some property which enables estimating of the influence of data perturbation on changes of goal function values. We present a method of generating a set of instances as the fist priority.

Let $\delta = \langle p, d, w\rangle$, where: $p = [p_{i,j}]_{n\times m}, d = [d_{i,j}]_{n\times m}$ and $w = [w_{i,j}]_{n\times m}$ are respectively: the matrix: of work execution times, completion times and penalty coefficient, will be some instances of (deterministic) data for **PFS** problem. By $D(\theta)$ we denote a set of data generated from θ through perturbation of time execution. This perturbation consists in changing of $p = [p_{i,j}]_{n\times m}$ elements into randomly determined values (i.e. numbers generated in accordance with certain distribution, for instance monotonous, etc.). Any element of $D(\theta)$ set takes form of $\langle p', d, w\rangle$ where perturbed elements of matrix $p' = [p'_{i,j}]_{n\times m}$, are determined randomly. Thus, $D(\theta)$ set includes instances of deterministic data for **PFS** problem, different from one another only by values of tasks' execution times.

Let $\boldsymbol{A} = \{\boldsymbol{TS}, \boldsymbol{TSF}, \boldsymbol{TSR}\}$, where ***TS***, ***TSF*** and ***TSR*** algorithms are: deterministic, fuzzy and probabilistic respectively. By π^A_δ we denote a solution (permutation) determined by $\boldsymbol{A}$ algorithm for δ data. The value of expression $W(\pi^A_\delta, \varphi)$ is cost (4) for an instance of deterministic φ data, when objects are executed in a sequence of (permutations) π^A_φ (i.e. in a sequence defined by $\boldsymbol{A}$ algorithm for δ) data. Then

$$\Delta(A, \delta, D(\delta)) = \frac{1}{|D(\delta)|}\sum_{\varphi\in D(\delta)} \frac{W(\pi^A_\delta, \varphi) - W(\pi^{TS}_\varphi, \varphi)}{W(\pi^{TS}_\varphi, \varphi)}.$$

We call a sustainability of π^A_δ solution determined by $\boldsymbol{A}$ algorithm on a set of $D(\delta)$ perturbed data. Determining π^{TS}_φ for a starting solution of ***TS*** algorithm π^A_φ was denoted and next

$$W(\pi^A_\delta, \varphi) - W(\pi^{TS}_\varphi, \varphi) \ge 0.$$

Thus, $\Delta(A, \delta, D(\delta)) \ge 0$. The value of expression $\Delta(A, \delta, D(\delta))$ is an average relative deviation of the best π^A_δ solution for the best set solutions, for every instances of perturbed data $\varphi \in D(\delta)$.

Let Ω be some set of deterministic instances for **PFS** problem. *Sustainability coefficient of* $\boldsymbol{A}$ *algorithm on a* Ω set we define as follows:

$$S(A, \Omega) = \frac{1}{|\Omega|}\sum_{\delta\in\Omega} \Delta(A, \delta, D(\delta)). \quad (8)$$

The smaller the coefficient, the more sustainable the solutions set by $\boldsymbol{A}$ algorithm i.e. small changes in value of data cause small changes of goal function value.

Computational experiments

Algorithms presented in this work were programmed in a C++ language. Computational experiments were carried out on a personal computer with a 2.2 GHz processor. Deterministic data were generated on the basis of 31 instances with a $(n \times m)$ size from 11×5 to 75×20 displayed on a OR Library [17]. For every instance (matrix of work execution times $p = [p_{i,j}]_{n\times m}$) there was matrix of terms $d = [d_{i,j}]_{n\times m}$ added and matrix of penalty coefficient $w = [w_{i,j}]_{n\times m}$. Elements of both matrices were generated randomly according to stable distribution adequately from a set $\{1, 2, \ldots, \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}\}$ and $\{1, 2, \ldots, 10\}$. Let Ω be a set of instances generated in accordance with this method. On the basis of instances from Ω the data was generated:

- fuzzy, for ***TSF*** algorithm (it is described precisely in chapter: **Fuzzy tasks times**),
- probabilistic, for ***TSR*** algorithm(it is described precisely in chapter: **Probabilistics tasks times**).

Next, for every instance of deterministic data $\delta \in \Omega$ there were 100 instances of perturbed data generated (elements of $D(\delta)$ set). Time perturbation p_{ij} $(i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, m)$ consists in replacing it with a new value randomly chosen in accordance with a normal distribution $N(p_{ij}, p_{ij}/10)$. In total, there were 3100 instances of perturbed data. With initiation of every algorithm there was a starting permutation $\pi = (1, 2, \ldots, n)$ assumed, and moreover:

- length of list of forbidden moves: n,
- number of algorithm's iterations: $n/2$ or n.

For every solution determined by $\boldsymbol{A} = \{\boldsymbol{TS}, \boldsymbol{TSF}, \boldsymbol{TSR}\}$ algorithm there was a percentage relative deviation determined:

$$\varepsilon(A) = \frac{W_A - W_{NEH}}{W_{NEH}} \cdot 100\%$$

where W_A is a value of solution determined by A algorithm, and W_{NEH} is a value determined by the best ***NEH*** construction algorithm [18] for a flow shop problem with a $C_{\max}$ criterion. The average values of these deviations are presented in Table 1. The values of solutions determined by each of the three presented ***TS*** algorithms are better than the solutions determined by ***NEH***. The best appeared to be a deterministic algorithm (average improvement 6.1%), the worst - probabilistic algorithm (average improvement 1.7%). Then, the sustainability coefficients (8) of deterministic ***TS***, fuzzy ***TSF*** and probabilistic ***TSR*** algorithm were determined. The calculations were made for the number of iterations of $n/2$ and n algorithm. For the biggest instances it is no more than 75 iterations. Owing to this procedure, the total time of calculations of the three tested algorithms does not exceed 5 minutes. With a larger (eg. n^2) number of iterations, the sustainability of separate algorithms increases slightly, whereas the time of calculations increases to a huge extent. Precise results are presented in Table 1.

Table 1
Average relative deviation $\varepsilon(A)$ and sustainability $S(A, \Omega)$ of algorithms: deterministic ***TS***, fuzzy ***TSF*** and probabilistic ***TSR***.

Iterations number	Average relative deviation			Sustainability		
	Algorithms					
	TS	***TSF***	***TSR***	***TS***	***TSF***	***TSR***
$n/2$	−4.8	−1.5	−0.6	6.1	2.8	6.0
n	−7.4	−4.8	−2.7	7.8	4.0	7.4
Average	**−6.1**	**−3.2**	**−1.7**	**6.9**	**3.9**	**6.4**

The most sustainable was a fuzzy ***TSF*** algorithm for which a sustainability coefficient $S(TSF, \Omega) = 3.9\%$. Algorithms: ***TS*** and ***TSR*** have similar sustainability coefficient $S(TS, \Omega) = 6.9\%$ and $S(TSR, \Omega) = 6.4\%$. However, they are far less sustainable than ***TSF*** algorithm.

While comparing the results, one may be surprised to discover that the increase of sustainability factor (i.e. worsening of sustainability) was accompanied by the increase of iteration number. It concerns both: deterministic and probabilistic algorithms. It results from the fact that better solutions (such were obtained in double increase of iteration number) are more vulnerable to any perturbation of data.

Case study

Investment task consists in realization of twelve residential buildings ($n = 12$). The buildings are characterized by a similar set of construction works creating an ordered element's ($m = 9$) sequence of works beginning from ground works and finishing with fit-out works. Basing on Standards of Outlays In-kind Catalogue the following times (matrix p) of work execution (in tenths of hours) were determined

7	8	7	7	7	8	7	7	6	7	5	4
8	11	8	9	9	11	8	9	8	9	8	8
8	11	10	9	9	11	10	9	11	9	9	9
7	8	7	7	8	8	7	7	8	8	8	7
6	7	7	7	7	7	7	7	7	7	8	15
11	14	11	13	13	14	11	13	14	13	14	8
9	14	9	11	10	13	9	11	8	10	11	9
4	8	6	7	5	7	7	8	9	9	9	5
6	9	5	9	7	5	8	9	8	7	7	7

During realization of the project it appeared that real times (matrix p' – perturbed data) differ from the originally established ones

10	8	7	8	7	8	6	7	6	8	5	5
8	10	8	9	8	11	9	9	8	10	8	7
9	11	12	9	9	10	10	11	11	9	8	9
7	8	9	7	7	8	7	10	8	8	9	7
6	6	7	7	7	8	7	7	8	7	8	16
11	12	11	13	12	14	11	13	14	12	14	7
11	14	9	10	10	13	9	10	8	10	12	9
4	7	6	7	4	7	7	6	9	9	10	5
4	9	4	9	6	3	8	6	6	7	6	6

In accordance with the description presented in chapter 5 the deadline times for works (matrix d) and weights of penalty function (matrix w) were determined. Next, on the basis of p matrix there was an instance of fuzzy and probabilistic data generated. With the use of deterministic (***TS***), fuzzy (***TSF***) and probabilistic (***TSR***) algorithms the three permutations (sequence of building's execution) were determined. For perturbed data (matrix p') and every permutation there was a value of penalty function calculated. The increase of penalty function value for fuzzy, probabilistic and deterministic algorithm was adequately: 3.6%, 9.1% and 17.4%. The most sustainable appeared to be a fuzzy ***TSF*** algorithm. The change of time for work execution (in relation to originals) caused the increase of penalty for not meeting the deadlines for work completion by 3.6%.

Summary

In this paper we present a certain problem of construction works scheduling. Because it is a difficult issue to define interchangeably the time of execution of individual works, thus, we model them with help of fuzzy numbers and random variables. In order to solve the problem we use an algorithm based on tabu search method. Computational experiments were carried out. The most sustainable appeared to be solutions determined by algorithm in which uncertain data are represented by fuzzy numbers.

The work was partially supported by the Polish Ministry of Science and Higher Education, grant No. N N514 470439.

References

[1] Zavadskas E.K., *Book review. Methods and models of research in construction projects engineering*, Journal of Business Economics and Management, 9, 3, 240–243, 2008.

[2] Zavadskas E.K., *History and evolving trends of construction colloquia on sustainability and operational research*, Technological and Economic Development of Economy, 14, (4), 578-592, 2008.

[3] Ustinovicius L., *Decision-support System for Determining the Efficiency of Investments in Construction: Summary of the research report presented for habilitation*, Vilnius, Technika, 2003.

[4] Dean B.C., *Approximation algorithms for stochastic scheduling problems*, PhD thesis, MIT, 2005.

[5] Vondrák J., *Probabilistic methods in combinatorial and stochastic optimization*, PhD, MIT, 2005.

[6] Ishibuschi H., Murata T., *Scheduling with Fuzzy Duedate and Fuzzy Processing Time*, in: Scheduling Under Fuzziness, R. Słowiński and M. Hapke [Eds.], Springer-Verlag, 2000, pp. 113–143.

[7] Ishii H., *Fuzzy combinatorial optimization*, Japanese Journal of Fuzzy Theory and Systems, 4, 1, 31–40, 1992.

[8] Garey M.R., Johnson D.S. and Seti R., *The complexity of flowshop and jobshop scheduling*, Mathematics of Operations Research, 1, 117–129, 1976.

[9] Osman I., Potts C., *Simulated Annealing for Permutation Flow-Shop Scheduling*, OMEGA, 17/6, 551–557, 1989.

[10] Ogbu F., Smith D., *The Application of the Simulated Annealing Algorithm to the Solution of the n/m/Cmax Flowshop Problem*, Computers & Operations Research, 17/3, 243–253, 1990.

[11] Bożejko W., Wodecki M., *Solving Flow Shop Problem by Parallel Simulated Annealing*, LNCS, Springer-Verlag, 2328, 236–244, 2002.

[12] Nowicki E., Smutnicki C., *A fast tabu search algorithm for the permutation flow-shop problem*, European Journal of Operational Research, 91, 160–175, 1996.

[13] Grabowski J., Wodecki M., *A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion*, Computers & Operations Research, 31, 1891–1909, 2004.

[14] Reeves C., *A Genetic Algorithm for Flowshop Sequencing*, Computers & Operations Research, 22/1, 5–13, 1995.

[15] Karp R.M., *Reducibility among combinatorial problems*, R.E. Miller and J.W. Thatcher [Eds.], Complexity of Computer Computation, Plenum Press, New York, 1972, pp. 85–104.

[16] Dubois D., Prade H., *Theorie des Possibilites. Applications a la representation des connaissances en informatique*, Paris: MASSON, 1988.

[17] OR Library http://people.brunel.ac.uk/mastjjb/jeb/orlib/flowshopinfo.html

[18] Navaz M., Enscoré E.E. and Ham I., *A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem*, OMEGA, 11/1, 91–95, 1983.