Blocks of Jobs for Solving Two-Machine Flow Shop Problem with Normal Distributed Processing Times

Wojciech Bożejko[1(✉)], Paweł Rajba[2], and Mieczysław Wodecki[3]

[1] Department of Control Systems and Mechatronics, Wrocław University of Science and Technology, Janiszewskiego 11/17, 50-372 Wrocław, Poland
wojciech.bozejko@pwr.edu.pl

[2] Institute of Computer Science, University of Wrocław, Joliot-Curie 15, 50-383 Wrocław, Poland
pawel@cs.uni.wroc.pl

[3] Department of Telecommunications and Teleinformatics, Wrocław University of Science and Technology, Janiszewskiego 11/17, 50-372 Wrocław, Poland
mieczyslaw.wodecki@pwr.edu.pl

Abstract. We consider strongly NP-hard problem of two-machine task scheduling with due dates and minimizing of the total weighted tardiness. Task execution times are random variables. We propose methods of intermediate review of solutions, the so-called 'block properties', which we use in the tabu search algorithm. From computational experiments carried out it follows that the use of blocks significantly speeds up calculations.

Keywords: Scheduling · Metaheuristics · Uncertain parameters

1 Introduction

In a two-machine flow shop problem with minimizing the sum of lateness costs (total tardiness, in short, F2T problem), each of the n tasks must be completed on the first machine and then on the second machine. The time of completing the tasks and the due dates (on the second machine) are given. Exceeding this due date will result in a penalty, which depends on the size of the delay (so called tardiness) and a fixed penalty factor (weight). The problem consists on determining the order of tasks (the same on both machines) which minimizes the sum of penalties. In the literature this problem is denoted by $F2||\sum w_iT_i$. It is a generalization of the NP-hard single-machine problem with the minimalization of sum of penalties for tardiness $1||\sum w_iT_i$ – a detailed description and algorithm of its solution is provided in the work of Bożejko et al. [4].

Á. Herrero et al. (Eds.): SOCO 2020, AISC 1268, pp. 289–298, 2021.
https://doi.org/10.1007/978-3-030-57802-2_28

There are relatively few papers devoted solely to the F2T problem and methods of solving it. Some theoretical results, as well as approximation algorithms are presented in the papers: Gupta and Harari [8], Lin [11], Bulfin and Hallah [7]. Various variants of this problem were also considered by Ahmadi et al. [10], Al-Salem et al. [1], Ardakan et al. [2] and Bank et al. [3]. Two-machine flow problem with $C_{\max}$ criterion (minimizing the end of execution time of all tasks, $F2||C_{\max}$) is a problem with polynomial computational complexity (Johnson's algorithm [9]).

The research of discrete optimization problems conducted for many years concerns the vast majority of deterministic models, in which the basic assumption is the uniqueness of all parameters. To solve these types of problems, which mostly belong to the class of strongly NP-hard problems, a number of effective approximate algorithms in which specific properties of problems are applied. However, in many areas of the economy we are dealing with random processes, e.g. transport, agriculture, trade, construction, etc. Effective management of such processes often leads to optimization models with random parameters. Already for the deterministic case solving these problems is very difficult, because they usually belong to the NP-hard class. The inclusion of parameter uncertainty in the model causes additional complications. Hence, the problems with random parameters are much less frequently studied. In this work we are considering a random problem with times of tasks execution. We present some properties of the problem (the so-called block elimination properties) accelerating the search of neighborhoods. Due to their implementation, it is possible to eliminate inferior solutions without having to calculate the value of the criterion function (intermediate review method). First, we will describe the case of a problem with deterministic task execution times, and then with the durations represented by random variables.

2 Two-Machine Problem with Due Dates

Two-machine flow shop scheduling problem with minimalization of total weighted tardiness (denoted by $F2||\sum w_i T_i$) can be formulated as follows:

F2T Problem. A set of tasks is given $\mathcal{J} = \{1, 2, \ldots, n\}$, and a set of machines $\mathcal{M} = \{1, 2\}$. A task $i \in \mathcal{J}$ consists of two operations O_{i1} and O_{i2}. An operation O_{ik} corresponds to performing a task i on a machine k. For a task $i \in \mathcal{J}$ we define:
p_{ik} – *execution time* (duration) of the operation O_{ik},
d_i – *requested completion time* (due date),
w_i – *weight of penalty function for exceeding the due date* (being tardy).

Each task should be executed on both machines and there must be fulfilled the following constraints:

(a) each task must be completed on the first and then on the second machine,
(b) the task cannot be interrupted,

(c) the request can be performed simultaneously on only one machine,
(d) the machine cannot perform more than one task at the same time,
(e) the task order, on both machines, must be the same.

For the fixed execution order on machines, let S_{ij} be the time of beginning of the operation O_{ij} ($i \in \mathcal{J},\ j = 1, 2$). From constraints (b) and (c) it follows, that $C_{ij} = S_{ij} + p_{ij}$ is the finishing time of the operation O_{ij}. These moments can be determined from the following recursive relationships:

$$C_{i,j} = \max\{C_{(i-1),j},\ C_{i,(j-1)}\} + p_{i,j}, \quad i = 1, 2,\ \ldots .n, \quad j = 1, 2, \tag{1}$$

with starting conditions $C_{0,j} = 0,\ j = 1, 2$ and $C_{i,0} = 0,\ i = 1, 2, \ldots, n$. By $\mathcal{C}_i = C_{i2}$ we denote finishing time of execution of the task i (operation O_{i2}). Therefore

$$T_i = \max\{0, \mathcal{C}_i - d_i\} \tag{2}$$

is *tardiness* of execution of the task i, $f_i = w_i \cdot T_i$ *penalty* for being tardy (*cost of a task execution*). If $T_i = 0$ than this task is called *early*, otherwise – *tardy*.

Any solution, i.e. the order in which the tasks are to be carried out (the same on both machines) can be represented by the permutation of tasks from the set $\mathcal{J}$. Let Π be a set of all such permutations. For any permutation $\pi = (\pi(1), \ldots, \pi(n)),\ \ \pi \in \Pi$, penalty for tardiness of tasks execution (cost)

$$\mathcal{T}(\pi) = \sum_{i=1}^{n} w_{\pi(i)} \cdot T_{\pi(i)}. \tag{3}$$

In the F2T problem under consideration, the order of tasks execution should be determined, which minimizes the sum of penalties for tardy tasks, i.e. optimal permutation $\pi^* \in \Pi$, for which

$$\mathcal{T}(\pi^*) = \min\{\mathcal{T}(\pi) :\ \pi \in \Pi\}. \tag{4}$$

In the introduction we wrote that the two-machine flow problem with the C_{max} criterion belongs to the P class. Johnson's algorithm [9] is used for solving this problem.

Any sequence of immediately following elements in we will call *sub-permutation.* If

$$\eta = (\pi(u), \pi(u+1), \ldots, \pi(v)),\ 1 \le u \le v \le n, \tag{5}$$

is a sub-permutation of a permutation π, then the cost of tasks execution from η

$$\mathcal{T}_\pi(\eta) = \sum_{i=u}^{v} (w_{\eta(i)} \cdot (\mathcal{C}_{\eta(i)} - d_{\eta(i)})), \tag{6}$$

where $\mathcal{C}_{\eta(i)}$ is a *finishing time* of execution of the task $\eta(i)$ in the permutation π. By $\mathcal{Y}(\eta)$ we denote a set of elements of sub-permutation η, i.e.

$$\mathcal{Y}(\eta) = \{\pi(u), \pi(u+1), \ldots, \pi(v)\}.$$

Let

$$\alpha = (1, 2, \ldots, a-1),\ \beta = (a, a+1, \ldots, b-1, b),\ \gamma = (b+1, \ldots, n), \tag{7}$$

where $1 \le a \le b \le n$ be sub-permutations in π.

Therefore permutation $\pi = (\alpha, \beta, \gamma)$ is a concatenation of three sub-permutations, and its cos

$$\mathcal{T}(\pi) = \mathcal{T}_\pi(\alpha) + \mathcal{T}_\pi(\beta) + \mathcal{T}_\pi(\gamma). \tag{8}$$

3 Random Task Execution Times

In this section we consider the probabilistic version of F2T problem – scheduling tasks on two machines with due dates. We assume that task execution times are independent random variables. Extensive review of methods and algorithms for solving optimization combinatorial problems with random parameters was presented by Vondrák in a monograph [12] and newer [13]. Some practical problems are also considered in the works of Bożejko et al. [6] and [5]. We will now introduce the necessary definitions and notions.

If X is a continuous random variable, we will use the following symbols later in this work:

F_X – cumulative distribution function of a random variable X,
$E(X)$ – expected value of a random variable X.

We consider, described in the previous chapter probabilistic version of the F2T problem, in which the task execution times $\tilde{p}_{ij}$ are independent random variables, and the remaining task parameters w_i and d_i $(i = 1, 2, \ldots, n)$ are deterministic. This problem we will briefly refer to as PF2T.

If tasks durations $\tilde{p}_{ij}$ are independent random variables, than for any tasks execution order $\pi \in \Pi$, a time of a task $\pi(k)$ finishing $\tilde{C}_{\pi(k)}$, tardiness $\tilde{T}_{\pi(k)} = \max\{0, \tilde{C}_{\pi(k)} - d_{\pi(k)}\}$ and cost function

$$\tilde{\mathcal{T}}(\pi) = \sum_{i=1}^{n} w_{\pi(i)} \cdot \tilde{T}_{\pi(i)}. \tag{9}$$

are also random variables.

There is a necessity in algorithms for solving optimization problems comparing the value of the criterion function for various acceptable solutions (e.g. permutation). In case this function is a random variable (9) we will be they used her expected value. Therefore, the following function will be used as comparative criteria for solutions:

$$\mathcal{L}(\pi) = E(\tilde{\mathcal{T}}(\pi)) = \sum_{i=1}^{n} w_{\pi(i)} \cdot E(\tilde{T}_{\pi(i)}). \tag{10}$$

In the rest of the work we present the methods for calculating the value of criterion function (10).

If $\beta = (\pi(a), \pi(a+1), \ldots, \pi(b))$, where $1 \leq a \leq b \leq n$ is a sub-permutation of a permutation $\pi \in \Pi$,

$$\mathcal{L}(\beta) = \sum_{i=a}^{b} w_{\pi(i)} \cdot E(\tilde{T}_{\pi(i)}). \tag{11}$$

is a cost of tasks execution of a sub-permutation β.

4 Blocks of Tasks

We consider a permutation $\pi \in \Pi$ – a solution of PF2T problem. If an expected value of the execution finishing time of a task $\pi(i)$, $E(\tilde{\mathcal{C}}_{\pi(i)}) \leq d_{\pi(i)}$ than this task $\pi(i)$ we call *early*, otherwise, if $E(\tilde{\mathcal{C}}_{\pi(i)}) > d_{\pi(i)}$, *late* (tardy).

Later in this chapter we present a method of constructing sub-permutations (called blocks) containing only early or tardy tasks.

4.1 Blocks of Early Tasks

Let a permutation $\pi \in \Pi$ define a sequence (7) of three sub-permutations, i.e. $\pi = (\alpha, \beta, \gamma)$. For tasks from sub-permutation $\beta = (a, a+1, \ldots, b-1, b)$ we assume the duration of the task on the machine

$$p_{ij} = E(\tilde{p_{ij}}),\ i \in \mathcal{J},\ j \in \mathcal{M}. \tag{12}$$

Then, we use the described in Sect. 2 Johnson's algorithm. In this way we set a new order of tasks from the set $\mathcal{Y}(\beta)$, i.e. sub-permutation

$$\beta' = (a', a'+1, \ldots, b'-1, b'). \tag{13}$$

We will call it *Johnson optimal*, in short *J-opt*. One can easily prove that this is the optimal order, due to the minimization of the expected date value of completion of all tasks in the set $\mathcal{Y}(\beta)$.

We consider permutations $\pi = (\alpha, \beta, \gamma)$ and $\pi' = (\alpha, \beta', \gamma)$. It's easy to show that if sub-permutation β' is *J-opt*, then the finishing time of the last task in β' is not greater than the finishing time of the last task in β.

Theorem 1. *Let β be J-opt sub-permutation in the permutation $\pi = (\alpha, \beta, \gamma)$, $\pi \in \Pi$. If permutation $\sigma = (\alpha, \delta, \gamma)$ was generated from π by changing the order of tasks in sub-permutation β, then expected value of the execution finishing time for any task from γ in the permutation σ is not less than the time of finishing of this task in the permutation π.*

The proof should use the assumption: sub-permutation β is $J - opt$.

Definition 1. *If all the tasks from J-opt sub-permutation β after insertion into the first position in β are on-time, then we call β* ***block of early tasks*** *(in short* ***T-block****).*

Theorem 2. *(Elimination T-block property) If a permutation σ was generated from $\pi \in \Pi$ by changing the order of tasks in a T-block, then*

$$\mathcal{L}(\sigma) \geq \mathcal{L}(\pi). \tag{14}$$

One should take advantage of *T-block* definition in proof.

Remark 1. *While generating new permutations from π one can omit these of them, which were generated by changing the order in any T-block. They do not give an improvement in the cost function value.*

4.2 Blocks of Tardy Tasks

Let a permutation of tasks order $\pi = (1, 2, \ldots, a-1, a, a+1, \ldots, b-1, b, b+1, \ldots, n) = (\alpha, \beta, \gamma)$, where $\alpha = (1, 2, \ldots, a-1)$, $\beta = (a, a+1, \ldots, b-1, b)$, $\gamma = (b+1, \ldots, n)$. We assume then that a sub-permutation β of all the tasks is tardy, and additionally

$$\forall i \in \beta,\ d_i < E(\tilde{\mathcal{C}}_{a-1} + \tilde{p}_{i,2}). \tag{15}$$

If $a = 1$ we assume $E(\tilde{\mathcal{C}}_{a-1}) = 0$. It follows from the above inequality that any task from the sub-permutation β inserted into the first position of β, i.e. position a is late (tardy). Let us assume, that the tasks from β fulfill inequalities (15). We generate two new permutations from π: $\pi' = (\alpha, \beta', \gamma)$ and $\pi'' = (\alpha, \beta'', \gamma)$, where $\mathcal{Y}(\beta) = \mathcal{Y}(\beta') = \mathcal{Y}(\beta''))$. We define sub-permutations β' and β'' occuring in both permutations as follows:

(a) in β' we set the order of the elements using the Johnson algorithm (it is therefore subpermutation *J-opt*),
(b) we construct the sub-permutation β'' by setting the tasks from the set $\mathcal{Y}(\beta)$, according to non-growing values $w_v/E((\tilde{p}_{v,1} + \tilde{p}_{v,2}))$, $v \in \mathcal{Y}(\beta)$.

Definition 2. *Sub-permutation β'' defined in (b) we call a **block of tardy tasks**, in short **D-block**, if*

$$(E(\tilde{\mathcal{C}}_{b''}) - E(\tilde{\mathcal{C}}_{b'}))/E(\tilde{\mathcal{C}}_{b''}) \leq \varphi,$$

where φ is a parameter which we assign experimentally.

While, $E(\tilde{\mathcal{C}}_{b''})$ and $E(\tilde{\mathcal{C}}_{b'})$ are expected values of finishing time of the last task in sub-permutation β' and β'', respectively, which were defined in (a) and (b).

A parameter φ (whose value is determined experimentally) is a measure enabling estimation of the difference between expected values of finishing times of tasks from $\mathcal{Y}(\beta)$ in order β'' and β'. For a small value of φ (e.g. 0.1) they differ only 'a little'. Then, β'' sub-permutation is quasi optimal for tasks from the set $\mathcal{Y}(\beta)$, both due to the expected value of the finishing time and the cost.

A *D-block* does not meet the elimination block property: '*reordering elements in block does not generate solutions with a smaller value of criterion function*'.

Despite this fact we will use them to eliminate certain solutions from neighborhood, due to its empirical advantage.

Any permutation can be partitioned into sub-permutation such that each is a *T-block* or a *D-block*. The algorithm for determining blocks is similar to the one presented in the paper [4] and has computational complexity $O(n^2)$.

5 Tabu Search Algorithm

Standard version of the tabu search method was used to solve the considered T2FS problem, with the neighborhood generated by insert type moves. The procedure of generating the environment uses blocks so that some elements are omitted without the necessity to calculate their criterion function value. In order to diversify the search process, a 'backtrack jump' mechanism was applied that resumes the process searches from remembered promising solutions. It is implemented through the introduction of the so-called long-term memory, abbreviated to LTM). A return jump (to the last saved LTM element) is executed in a case where through a certain number of iterations there is no improvement of the best solution. The algorithm terminates after a fixed number of iterations.

6 Random Tasks Execution Times

Let $\delta = (\tilde{p}_{ij}, w_i, d_i),\ i = 1, 2, \ldots, n,\ j = 1, 2$ be a data instance for the PF2T problem, where tasks execution times $\tilde{p}_{ij}$ are independent random variables with normal distribution, i.e. $\tilde{p}_{ij} \sim N(p_{ij}, \lambda p_{ij})$, and λ a fixed parameter. For the simplification, let's assume that the order of performing tasks (the same on both machines)

$$\beta = (1, 2, \ldots, n). \tag{16}$$

To calculate the value of the cost function

$$\mathcal{L}(\beta) = E(\tilde{\mathcal{T}}(\beta)) = \sum_{i=1}^{n} w_{\beta(i)} \cdot E(\tilde{T}_{\beta(i)}), \tag{17}$$

where $\tilde{T}_{\beta(k)} = \max\{0, \tilde{\mathcal{C}}_{\beta(k)} - d_{\beta(k)}\}$, it is necessary to determine tasks execution finishing times $\tilde{\mathcal{C}}_{\beta(k)}$. We introduce additional values:

$$C'_{i,j} = \begin{cases} \sum_{k=1}^{i} p_{i,j}, & \text{for } j = 1, \\ C'_{i,j-1} + p_{i,j}, & \text{for } i = 1, j = 2, \\ \max\{C'_{i,j-1}, C'_{i-1,j}\} + p_{i,j}, & \text{for } i > 1, j = 2. \end{cases}$$

Finally random variable representing the time of finishing of the i-th task

$$\tilde{\mathcal{C}}'_i \sim N(C'_{i2},\ \lambda\sqrt{C'_{i2}}).$$

Let

$$\mu_i = p_{1,1} + p_{2,1} + \ldots, + p_{i,1} + p_{i,2} \text{ and } \sigma_i = \lambda\sqrt{p_{1,1}^2 + p_{2,1}^2 + \ldots, + p_{i,1}^2 + p_{i,2}^2}.$$

When calculating the expected value $E(\tilde{T}_i)$ appearing in the definition of the criterion function (17), we will use the following theorem.

Theorem 3. *If tasks duration are independent random variables with normal distribution* $\tilde{p}_{ik} \sim N(p_{ik}, \lambda \cdot p_{ik})$ $(i = 1, 2, \ldots, n,\ k = 1, 2)$ *then an expected value of tardiness of the task* $i \in \mathcal{J}$

$$E(\tilde{T}_i) = (1 - F_{\tilde{\mathcal{C}'}_i(d_i)}) \left(\frac{\sigma_i}{\sqrt{2\pi}} e^{\frac{-(d_i-\mu_i)^2}{2\sigma_i^2}} + (\mu_i - d_i) \left(1 - F_{N(0,1)}(\frac{d_i - \mu_i}{\sigma_i}) \right) \right).$$

The proof of this theorem is similar to the one given in Bożejko et al. [6].

7 Computational Experiments

Computational experiments were carried out on two versions of the tabu search algorithm for solving the PF2T probabilistic problem:

1. `PTS` – an algorithm with neighborhood generated by swap moves,
2. `PTR+b` – an algorithm with elimination block properties application.

The main goal of the experiments was to examine the stability of algorithms, i.e. the resistance of solutions to random data disturbances (times of operations). The exact algorithm stability was described in the paper of Bożejko et al. [6]. The starting solution for both algorithms was the natural permutation $\pi = (1, 2, \ldots, n)$, and in addition: the length of the list of tabu moves: n, number of algorithm iterations: $2n$. The algorithms have been implemented in C++ and run on a PC with a 2.8 GHz clock.

Because in the literature there are no examples of tests for the problem under consideration, for the needs of execution computational experiments were generated randomly. Times for completing tasks on individual machines were randomly designated, in accordance with uniform distribution from the set $\{1, 2, \ldots, 99\}$, and weights the penalty function w_i, from the set $\{1, 2, \ldots, 9\}$. The values requested deadlines for completing tasks have been set based on two parameters: T – *latency factor* and R – *timeliness range*. These terms, according to the uniform distribution, were randomly selected from the range $[P(1-T-R/2), (1-T+R/2)]$, where $P = \sum_{i=1}^{n} \sum_{j=1}^{2} p_{i,j}$. Test examples were generated for each pair of parameter values $T = 0.2, 0.4$ and $R = 0.2, 0.4$ (the larger the coefficients, the more difficult the generated examples). There are four such pairs in total. The examples were generated for the number of tasks $n = 100$ and 500. For each pair of T and R values 25 examples were generated. Ultimately, 200 examples were used for the performed computational experiments whose collection (the so-called *deterministic data*) is denoted by Ω. Then, for each example (p_{ij}, w_i, d_i), $i = 1, 2, \ldots, n$, $j = 1, 2$ of the deterministic problem two examples of probabilistic data were determined $(\tilde{p}_{ij}, w_i, d_i)$, where the operation duration was a random variable with a normal distribution $\tilde{p}_{ij} \sim N(d_i, \lambda \cdot p_{ij})$ for $\lambda = 0.02, 0.05$. The collection of this data (the so-called *probabilistic data*) is denoted by $\widetilde{\Omega}$. To examine the stability of probabilistic algorithms, a disturbed data set $\Omega^{\approx}$ was generated. For each example of deterministic data (p_{ij}, w_i, d_i) 50 examples of disturbed data were generated. The disorder consists in changing the duration of operation p_{ij} into randomly designated value, according to the distribution of $N(p_{ij}, \lambda \cdot p_{ij})$. In total, 10,000 examples were given. They were then solved by the tabu search algorithm. The results obtained were the basis for determination of the stability coefficient of probability algorithms `PTS` and `PTS+b`. Aggregate results are included in the Table 1.

Table 1. Stability coefficient of probabilistic algorithms.

Instance	n	$\lambda = 0.02$		$\lambda = 0.05$	
		PTS	PTS+b	PTS	PTS+b
$T = 0.2, R = 0.2$	100	3.41	3.37	2.24	2.27
$T = 0.2, R = 0.4$	100	3.82	3.87	3.06	2.99
$T = 0.4, R = 0.2$	100	4.78	4.77	4.02	3.99
$T = 0.4, R = 0.4$	100	5.12	5.01	3.52	3.35
$T = 0.2, R = 0.2$	500	4.68	4.55	4.79	4.66
$T = 0.2, R = 0.4$	500	5.34	5.93	6.13	6.31
$T = 0.4, R = 0.2$	500	7.11	6.58	8.09	8.01
$T = 0.4, R = 0.4$	500	10.28	9.54	12.64	10.07
Average		**5.57**	**4.80**	**5.56**	**4.32**

The main purpose of the carried out experiments was to examine individual stability of algorithms, i.e. the robustness of solutions determined by these algorithms for random changes (disturbances) of parameters. Among the probabilistic algorithms tested, he proved to be more stable the 'with blocks' algorithm `PTS+b`. Its stability factor is 4.56. That is, the data disorder (according to the described random procedure) causes average relative deterioration criterion (in relation to the best solution of this example) of 5.56%. The `PTS+b` algorithm stability factor is 5.57. In addition, it turned out that the use of blocks in a probabilistic algorithm resulted in a shortening of the average calculation time by about 30%. The results obtained prove the high efficiency of the blocks.

8 Conclusions

The paper examines the problem of scheduling tasks on two machines, in which the times of task execution are random variables. Blocks of tasks were introduced to eliminate the use of solutions with the environment generated by swap movements that require in the taboo algorithm search. Computational experiments were carried out in order to study the impact of blocks on computation and analize the times of designated solutions. The results obtained are clearly available, the use of blocks significantly reduces calculation time and improves the stability of solutions. Application of elements of probability in the adaptation of tabu search methods allows one to solve uncertain data problems. These are very difficult optimization problems, much better describing reality than deterministic models.

Acknowledgments. This work was partially funded by the National Science Centre of Poland, grant OPUS no. 2017/25/B/ST7/02181 and a statutory subsidy 049U/0032/19.

References

1. Al-Salem, M., Valencia, L., Rabadi, G.: Heuristic and exact algorithms for the two-machine just in time job shop scheduling problem. Math. Prob. Eng. **5**, 1–11 (2016)
2. Ardakan, M., Beheshti, A., Hamid Mirmohammadi, S., Ardakani, H.D.: A hybrid meta-heuristic algorithm to minimize the number of tardy jobs in a dynamic two-machine flow shop problem. Numer. Algebra Control Optim. **7**(4), 465–480 (2017)
3. Bank, M., Fatemi, S., Ghomi, M.T., Jolai, F., Behnamian, J.: Two-machine flow shop total tardiness scheduling problem with deteriorating jobs. Appl. Math. Model. **36**(11), 5418–5426 (2012)
4. Bożejko, W., Grabowski, J., Wodecki, M.: Block approach tabu search algorithm for single machine total weighted tardiness problem. Comp. Ind. Eng. **50**(1–2), 1–14 (2006)
5. Bożejko, W., Hejducki, Z., Wodecki, M.: Flowshop scheduling of construction processes with uncertain parameters. Arch. Civil Mech. Eng. **19**, 194–204 (2019)
6. Bożejko, W., Rajba, P., Wodecki, M.: Stable scheduling of single machine with probabilistic parameters. Bull. Pol. Acad. Sci. Tech. Sci. **65**(2), 219–231 (2017)
7. Bulfin, R.L., M'Hallah, R.: Minimizing the weighted number of tardy jobs on two-machineflow shop. Comput. Oper. Res. **30**, 1887–1900 (2003)
8. Gupta, J.N.D., Hariri, A.M.A.: Two-machine flowshop scheduling to minimize the number of tardy jobs. J. Oper. Res. Soc. **48**, 212–220 (1997)
9. Johnson, S.M.: Optimal two- and three-stage production schedules with setup times included. Naval Res. Logist. Q. **1**, 61–68 (1954)
10. Ahmadi-Darani, M., Moslehi, G., Reisi-Nafchi, M.: A two-agent scheduling problem in a two-machine flowshop. Int. J. Ind. Eng. Comput. **9**(3), 289–306 (2018)
11. Lin, B.M.T.: Scheduling in the two-machine flowshop with due date constraints. Int. J. Prod. Econ. **70**, 117–123 (2001)
12. Vondrák, J.: Probabilistic methods in combinatorial and stochastic optimization. PhD, MIT (2005)
13. Cai, X., Wu, X., Zhou, X.: Optimal Stochastic Scheduling. Springer, New York (2014)