UDC 519.852+519.687.1

# The minimization of exact total weighted completion time in the preemptive scheduling problem by subsequent length-equal job importance growth

V. V. Romanuke
*Polish Naval Academy, Poland*
*romanukevadimv@gmail.com*

For the preemptive scheduling problem in case of subsequent job importance growth, it is studied whether the optimal schedule might be found faster within an exact model. It is ascertained that when the number of jobs up to six (except for the case of four jobs) and there is no randomness in problem forming, a little advantage of weight-descending job order exists only on average. As the number of jobs increases, the advantage of either weight-descending or weight-ascending job order becomes more certain. When priority weights are formed randomly, weight-descending job order is expected to be faster than weight-ascending.

***Key words:*** *optimal schedule, job order, preemption, total weighted completion time, exact solution, computation time gain.*

Теорія розкладів є важливою галуззю прикладної математики, спрямованою на організацію багатоетапних процесів компонування, виробництва, будівництва, диспетчеризації, обчислень тощо. Одним з головних критеріїв є мінімізація загального зваженого часу завершення. Існує клас задач планування з перемиканнями, у яких виконання одного завдання може бути перервано на користь виконання іншого завдання. У цьому класі існує підклас задач, у якому значущість наступних завдань зростає. Такі задачі планування виникають у системах, чий розвиток стає більш складним разом зі зростаючими витратами для підтримки цього розвитку. Коли кількість завдань сягає кількох сотень, такі задачі можуть розв'язуватися за допомогою евристик. Але знаходження точного мінімального загального зваженого часу завершення є цілком можливим для випадків з кількома завданнями, хоча це і потребує значно довших обчислень. Однак, оскільки евристики можуть дати лише наближені розклади, точні розклади для короткострокових задач планування все ще представляють інтерес. Тому метою є з'ясувати, чи оптимальний розклад міг би бути знайдений швидше у рамках точної моделі. Встановлюється, що за числа завдань до шести, без випадковостей у формуванні задачі, за виключенням випадку з чотирма завданнями, слабка перевага порядку завдань зі спадаючими вагами існує лише у середньому. Зі зростанням кількості завдань перевага порядку завдань або зі спадаючими вагами, або зі зростаючими вагами стає більш чіткою. Коли ваги пріоритетів сформовані випадково, очікується, що саме порядок завдань зі спадаючими вагами буде швидшим.

***Ключові слова:*** *оптимальний розклад, порядок завдань, перемикання, загальний зважений час завершення, точний розв'язок, виграш у часі обчислення.*

В задаче планирования с переключениями при возрастании значимости последующих заданий изучается вопрос о том, может ли оптимальное расписание быть найдено быстрее в пределах точной модели. Устанавливается, что с числом заданий до шести, без случайностей в формировании задачи, за исключением случая с четырьмя заданиями, слабое преимущество порядка заданий с убывающими весами существует только в среднем. С возрастанием количества заданий преимущество порядка заданий либо с убывающими весами, либо с возрастающими весами стаёт более чётким. Когда веса приоритетов сформированы случайно, ожидается, что именно порядок заданий с убывающими весами будет быстрее, чем с возрастающими весами.

***Ключевые слова:*** *оптимальное расписание, порядок заданий, переключение, общее взвешенное время завершения, точное решение, выигрыш во времени вычисления.*

## The preemptive scheduling problem by subsequent job importance growth

The scheduling theory is a quite important field of applied mathematics helping in organizing multistep processes of assembling, manufacturing, building, dispatching, computing, etc. [1, 2]. Given a number $N$ of jobs, where $N \in \mathbb{N} \setminus \{1\}$ and each of them has its own importance designated as a weight, one of main criteria is minimizing the total weighted completion time (TWCT) [2, 3]. There exists a class of preemptive scheduling problems (PESPs), wherein a job can be interrupted in favor of another job [3]. A subclass of this class contains problems in which importance of subsequent jobs grows. Such PESPs arise in systems whose development becomes more complicated along with its growing costs. When $N$ is of order of hundreds, PESPs can be solved only by using some heuristics [1, 2, 3]. Meanwhile, finding the exact minimal TWCT is surely possible for a few jobs, although it requires much longer computation time compared to heuristics [3, 4]. However, whereas heuristics may give only approximate solutions, exact solutions for short-termed scheduling are still a matter of interest.

**An approach to find the exact minimal TWCT by the Boolean linear programming model**

Let job $n$ have a processing period (PP) $H_n$ (job $n$ is divided into $H_n$ equal parts), a release date (RD) $r_n$, and a priority weight (PW) $w_n$, $n=\overline{1,N}$. Vectors of PPs, PWs, and RDs are

$$\mathbf{H}=\left[H_n\right]_{1\times N}\in\mathbb{N}^N,\ \ \mathbf{W}=\left[w_n\right]_{1\times N}\in\mathbb{N}^N,\ \ \mathbf{R}=\left[r_n\right]_{1\times N}\in\mathbb{N}^N, \tag{1}$$

respectively, where $r_n$ is the time moment, at which job $n$ becomes available for processing, and

$$\exists n_1\in\left\{\overline{1,N}\right\}\ \text{ such that }\ r_{n_1}=1,\ 1+\sum_{n=1}^{k}\bar{H}_n\geqslant\bar{r}_{k+1}\quad\forall k=\overline{1,N-1} \tag{2}$$

by having sorted components of $\mathbf{R}$ in ascending order to vector $\bar{\mathbf{R}}=\left[\bar{r}_n\right]_{1\times N}$, whereupon vector $\mathbf{H}$ becomes respectively sorted after $\bar{\mathbf{R}}=\left[\bar{r}_n\right]_{1\times N}$ to $\bar{\mathbf{H}}=\left[\bar{H}_n\right]_{1\times N}$. The goal is to minimize the TWCT, i. e. to schedule the jobs so that sum

$$\sum_{n=1}^{N}w_n\theta\left(n;H_n\right) \tag{3}$$

would be minimal, where job $n$ is completed after moment $\theta\left(n;H_n\right)$, which is

$$\theta\left(n;H_n\right)\in\left\{\overline{1,T}\right\}\ \text{ by }\ T=\sum_{n=1}^{N}H_n\,. \tag{4}$$

Let $x_{nh_nt}=1$ if the $h_n$-th part of job $n$ is assigned to time moment $t$; otherwise, $x_{nh_nt}=0$. The respective triple-indexed weights are calculated as follows:

$$\lambda_{nh_nt}=0\ \text{ by }\ r_n-1+h_n\leqslant t\leqslant T-H_n+h_n\quad\forall h_n=\overline{1,H_n-1} \tag{5}$$

and

$$\lambda_{nh_nt}=\alpha\ \text{ by when double inequality in (5) is not true,} \tag{6}$$

where $\alpha>0$ is a sufficiently great integer (similar to the meaning of infinity), e. g., $\alpha=\sum_{n=1}^{N}\sum_{t=1}^{T}w_nt$;

$$\lambda_{nH_nt}=w_nt\ \text{ by }\ r_n-1+H_n\leqslant t\leqslant T \tag{7}$$

and

$$\lambda_{nH_nt}=\alpha\ \text{ by when double inequality in (7) is not true.} \tag{8}$$

Denote an aggregate of all variables by $X=\left\{\left\{\left\{x_{nh_nt}\right\}_{n=1}^{N}\right\}_{h_n=1}^{H_n}\right\}_{t=1}^{T}\in A$, where $A$ is a set of all possible aggregates. The factual goal is to find such an aggregate

$$X^*=\left\{\left\{\left\{x^*_{nh_nt}\right\}_{n=1}^{N}\right\}_{h_n=1}^{H_n}\right\}_{t=1}^{T}\in\arg\min_{X\in A}\sum_{n=1}^{N}\sum_{h_n=1}^{H_n}\sum_{t=1}^{T}\lambda_{nh_nt}x_{nh_nt} \tag{9}$$

by constraints which constitute set $A$ (an integer binary lattice):

$$x_{nh_nt}\in\left\{0,1\right\}\ \text{ by }\ n=\overline{1,N}\ \text{ and }\ h_n=\overline{1,H_n}\ \text{ and }\ t=\overline{1,T}\,, \tag{10}$$

$$\sum_{t=1}^{T}x_{nh_nt}=1\ \text{ by }\ n=\overline{1,N}\ \text{ and }\ h_n=\overline{1,H_n}\,,\ \ \sum_{n=1}^{N}\sum_{h_n=1}^{H_n}x_{nh_nt}=1\ \text{ by }\ t=\overline{1,T}\,, \tag{11}$$

$$\sum_{j=t+1}^{T}\sum_{h_n=1}^{H_n-1}x_{nh_nj}+H_nx_{nH_nt}\leqslant H_n\ \text{ by }\ n=\overline{1,N}\ \text{ and }\ t=\overline{1,T-1}\,. \tag{12}$$

Formulae (1) — (12) is the Boolean linear programming model (BLPM) [4] allowing to find the optimal job schedule $\mathbf{S}^*=\left[s_t^*\right]_{1\times T}$ by $s_t^*\in\left\{\overline{1,N}\right\}$ for every $t=\overline{1,T}$ and its exact minimal TWCT [3]

$$\rho^*\left(N\right)=\sum_{n=1}^{N}\sum_{h_n=1}^{H_n}\sum_{t=1}^{T}\lambda_{nh_nt}x^*_{nh_nt} \tag{13}$$

for those $N$ jobs. However, problem (9) by the BLPM is NP-hard [1, 2, 4]. It becomes hardly tractable for a few tens of jobs. So, reducing the BLPM computation time even for a few jobs is very important.

**An example of that the job order into the BLPM matters**

Consider a PESP by subsequent job importance growth of eight jobs:

$$\mathbf{H}=\left[H_n\right]_{1\times 8}=\left[2\right]_{1\times 8},\ \ \mathbf{W}=\left[w_n\right]_{1\times 8}=\left[1\ \ 3\ \ 7\ \ 16\ \ 33\ \ 44\ \ 55\ \ 67\right],\ \ \mathbf{R}=\left[r_n\right]_{1\times 8}=\left[n\right]_{1\times 8}. \quad (14)$$

Here the total processing time is $T=16$, and the optimal schedule given by the BLPM (1) — (12) is

$$\mathbf{S}^*=\left[s_t^*\right]_{1\times 16}=\left[1\ \ 1\ \ 3\ \ 3\ \ 5\ \ 5\ \ 7\ \ 7\ \ 8\ \ 8\ \ 6\ \ 6\ \ 4\ \ 4\ \ 2\ \ 2\right]. \quad (15)$$

The exact minimal TWCT by schedule (15) is 2138. However, this problem might be stated as

$$\mathbf{H}=\left[H_n\right]_{1\times 8}=\left[2\right]_{1\times 8},\ \ \mathbf{W}=\left[w_n\right]_{1\times 8}=\left[67\ \ 55\ \ 44\ \ 33\ \ 16\ \ 7\ \ 3\ \ 1\right],\ \ \mathbf{R}=\left[r_n\right]_{1\times 8}=\left[8-n+1\right]_{1\times 8}. \quad (16)$$

Then, obviously, the optimal schedule given by the BLPM (1) — (12) remains the same, that is

$$\mathbf{S}^*=\left[s_t^*\right]_{1\times 16}=\left[8\ \ 8\ \ 6\ \ 6\ \ 4\ \ 4\ \ 2\ \ 2\ \ 1\ \ 1\ \ 3\ \ 3\ \ 5\ \ 5\ \ 7\ \ 7\right] \quad (17)$$

giving the exact minimal TWCT of 2138, but it is obtained at least by 25 % faster than schedule (15). Thus, the job order in the BLPM (which uses the branch-and-bound approach for solving [4]) matters.

**The object of studying**

Taking into account that the job order in the BLPM may determine the computation time, the purpose is to study whether the optimal schedule is found faster in the PESP by subsequent length-equal job importance growth (SLEJIG) when the job order is weight-descending (just like in the example above). The PP will be set at 2 for simplification. To achieve the goal, the three following tasks are to be accomplished:

1. To compare computation times of weight-descending job order (WDJO) and weight-ascending job order (WAJO) for PESPs with trivially increasing PWs and RDs. RDs will be set trivially as

$$\mathbf{R}=\left[r_n\right]_{1\times N}=\left[n\right]_{1\times N}\ \text{ for WAJO and }\ \mathbf{R}=\left[r_n\right]_{1\times N}=\left[N-n+1\right]_{1\times N}\ \text{ for WDJO.} \quad (18)$$

2. To compare computation times of WDJO and WAJO for PESPs by SLEJIG, whose PWs are formed randomly by definite statistical laws along with (18) and sorted. The comparison will be a ratio

$$\beta\left(N\right)=100\cdot\frac{\tau_{As}\left(N\right)-\tau_{Des}\left(N\right)}{\tau_{Des}\left(N\right)}, \quad (19)$$

where $\tau_{As}\left(N\right)$ and $\tau_{Des}\left(N\right)$ are averages of computation times for WAJO and WDJO, respectively.

3. To finally conclude on the statistical results and find out regularities/laws, if any, of solving PESPs by SLEJIG with WDJO/WAJO by the BLPM. Then practical recommendations are to be given.

**PESPs with trivially increasing PWs and RDs**

Here, two versions of PESPs with WAJO and WDJO are given as

$$H_n=2,\ \ r_n=n\ \ \text{by}\ \ w_n=n\ \text{(WAJO) and}\ \ r_n=N-n+1\ \ \text{by}\ \ w_n=N-n+1\ \ \forall n=\overline{1,N}\ \text{(WDJO),} \quad (20)$$

and

$$H_n=2,\ \ r_n=n\ \ \text{by}\ \ w_n=2n-1\ \ \text{(WAJO)}$$

$$\text{and}\ \ r_n=N-n+1\ \ \text{by}\ \ w_n=2N-2n+1\ \ \forall n=\overline{1,N}\ \ \text{(WDJO),} \quad (21)$$

respectively, for $N=\overline{2,10}$. Fig. 1 shows ratio (19) for cases (20) and (21), whence it follows that WDJO is definitely computed faster for $N\in\left\{2,3,6\right\}$. This particular conclusion is confirmed by disclosing the averages — see Fig. 2, wherein two bunches of ratios (19) for cases (20) and (21) obtained separately in the repetitious solutions of the same problem (9) by BLPM (1) — (12) are really tight, especially when $N$ increases. On average, only PESPs with nine jobs are solved slower by WDJO than by WAJO.

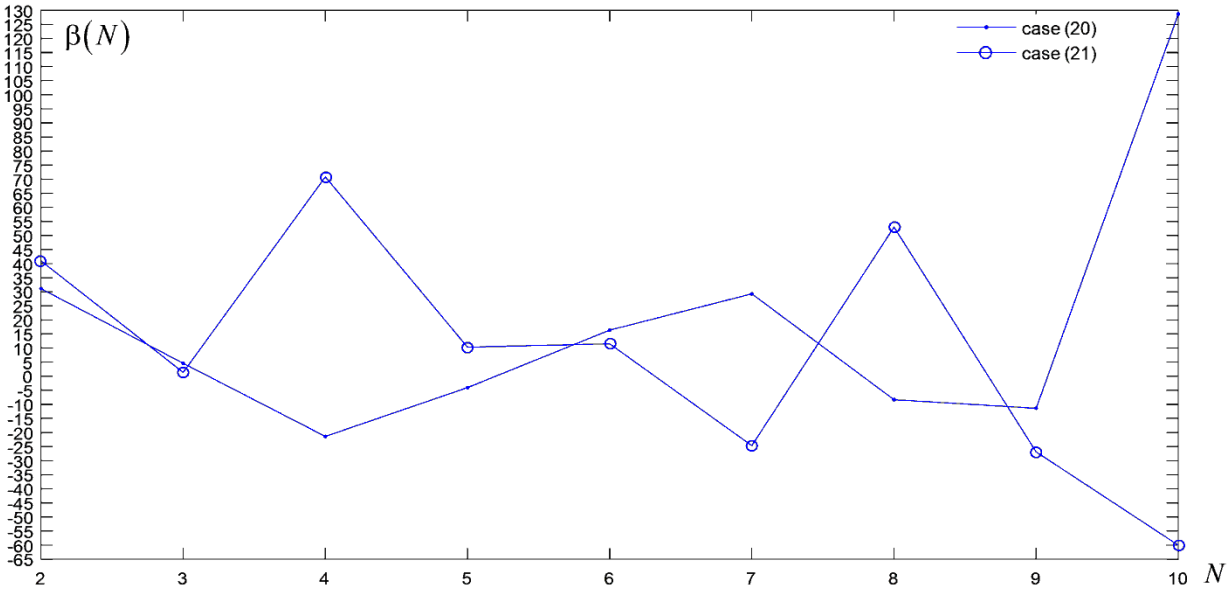


*Fig. 1. Ratio (19) for cases (20) and (21): on average, PESPs by WDJO are 13.4 % solved faster than by WAJO*

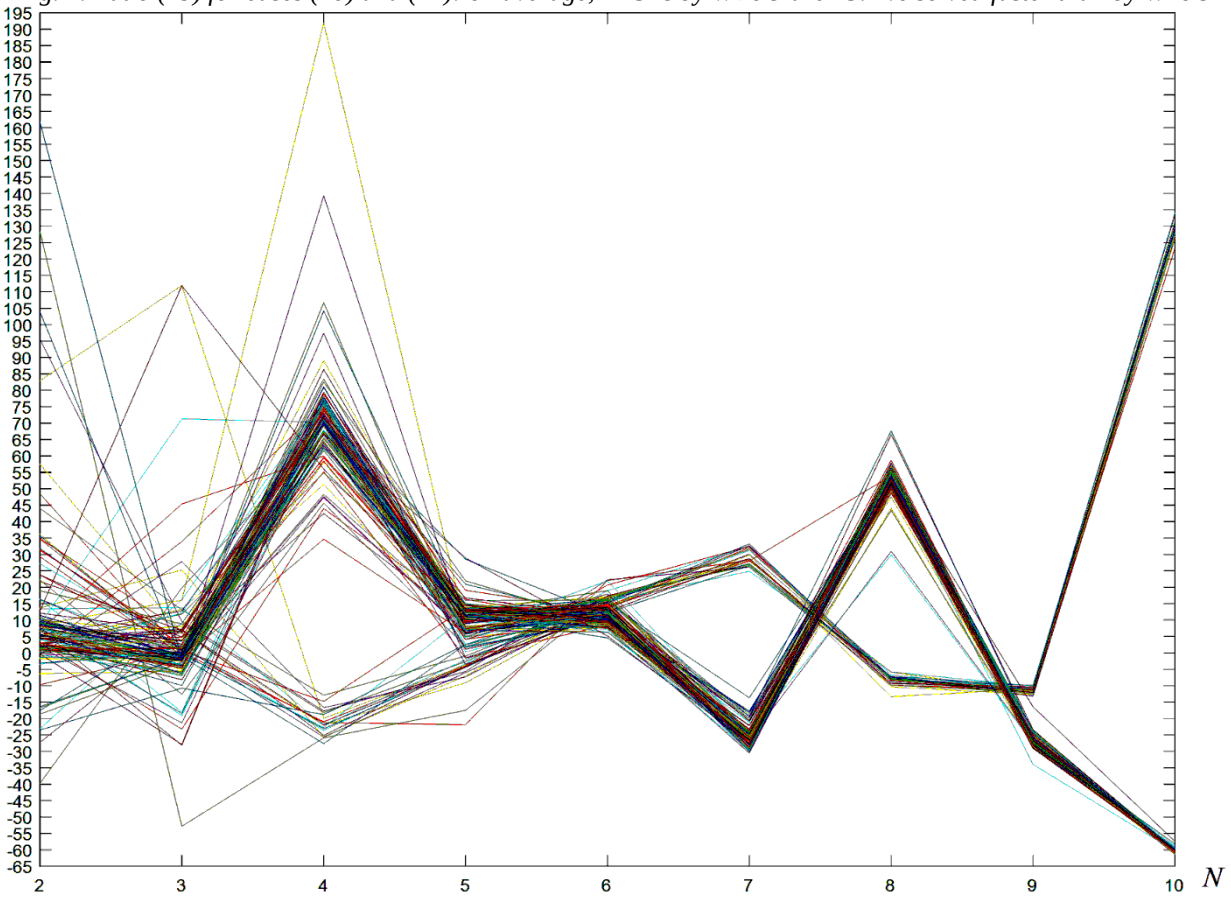


*Fig. 2. Two bunches of ratios (19) for cases (20) and (21) obtained separately in the repetitious solutions of the same problem (9) by BLPM (1) — (12)*

Obviously, cases (14) and (16), (20) and (21) do not confirm that PESPs by WDJO are solved faster. Moreover, Fig. 1 and 2 prove that WAJO by SLEJIG is preferable for 4, 5, 8, 9 jobs by case (20), where PWs are the most compact, and for 7, 9, 10 jobs by case (21), where PWs are slightly loose. Nevertheless, the averaged time-advantage of WDJO is an evidence of that the job order in the BLPM might be optimized for other cases of PWs generated by the certain laws (e. g., see [5]).

**PESPs whose PWs are formed randomly along with (18) and sorted**

Let PWs for WAJO be generated by the following law:

$$H_n = 2,\ \ r_n = n \ \text{ by } \ w_n = w_n^{\langle 0\rangle} + \gamma d_n,\ \ \gamma \in \{0,1\},\ \ w_n^{\langle 0\rangle} = \psi\left(N\zeta + 1\right) \ \ \forall n = \overline{1,\ N}$$

$$\text{and } \ w_{l-1}^{\langle 0\rangle} \leqslant w_l^{\langle 0\rangle} \ \ \forall l = \overline{2,\ N} \ \text{ but } \ \exists l_* \in \left\{\overline{2,\ N}\right\} \ \text{ such that } \ w_{l_*-1}^{\langle 0\rangle} < w_{l_*}^{\langle 0\rangle}, \tag{22}$$

where $\zeta$ is a pseudorandom number drawn from the standard uniform distribution on the open interval $(0;1)$, function $\psi(\xi)$ returns the integer part of number $\xi$, and

$$d_1 = 1,\ \ d_2 = 3,\ \ d_k = \psi\left(d_{k-2} + d_{k-1} + \sigma_i k\right),\ \ k = \overline{3,\ N} \ \text{ by } \ \sigma_i = i/8,\ \ i = \overline{0,8}. \tag{23}$$

PWs for WDJO are generated similarly to (22) by (23):

$$H_n = 2,\ \ r_n = N - n + 1 \ \text{ by } \ w_n = w_n^{\langle 0\rangle} + \gamma d_{N-n+1},\ \ \gamma \in \{0,1\},\ \ w_n^{\langle 0\rangle} = \psi\left(N\zeta + 1\right) \ \ \forall n = \overline{1,\ N}$$

$$\text{and } \ w_{l-1}^{\langle 0\rangle} \geqslant w_l^{\langle 0\rangle} \ \ \forall l = \overline{2,\ N} \ \text{ but } \ \exists l_* \in \left\{\overline{2,\ N}\right\} \ \text{ such that } \ w_{l_*-1}^{\langle 0\rangle} > w_{l_*}^{\langle 0\rangle}. \tag{24}$$

Fig. 3 shows ratio (19) for 10 cases by (22) — (24), whence it follows that WDJO is computed faster for $N \in \{4, 6, 10\}$. It is preferably to use WAJO for PESPs with three jobs. The average computation time gain by ratio (19) is 15.9 % for PESPs with 10 jobs. This gain is over 28.7 % for PESPs with four and six jobs. The lowest gain has been revealed for PESPs with five and eight jobs. Moreover, disadvantage of WDJO has occurred for every studied number of jobs (see Fig. 4). The averaged advantage of WDJO is nonetheless observed, although it doesn't grow as the number of jobs increases.

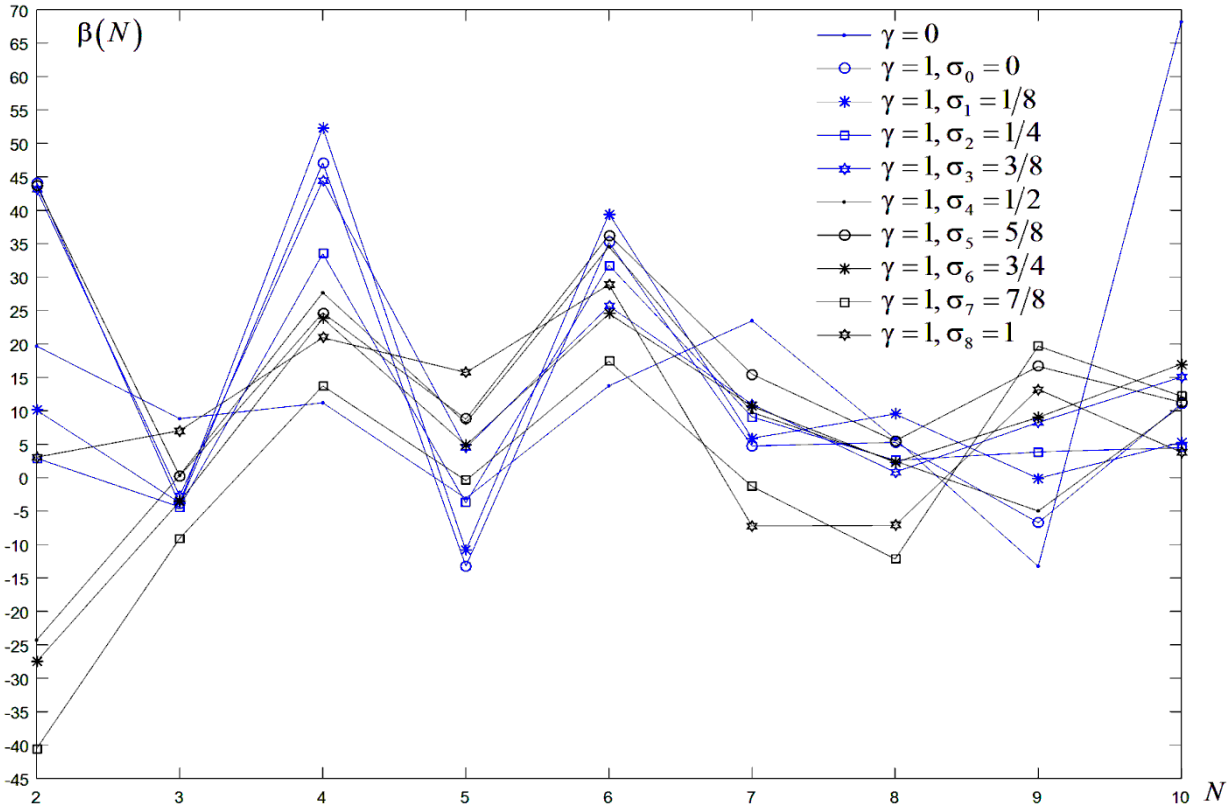


*Fig. 3. Ratios (19) for 10 cases by (22) — (24), wherein PESPs with the even number of jobs seem to be better to be solved by WDJO when using BLPM (1) — (12)*

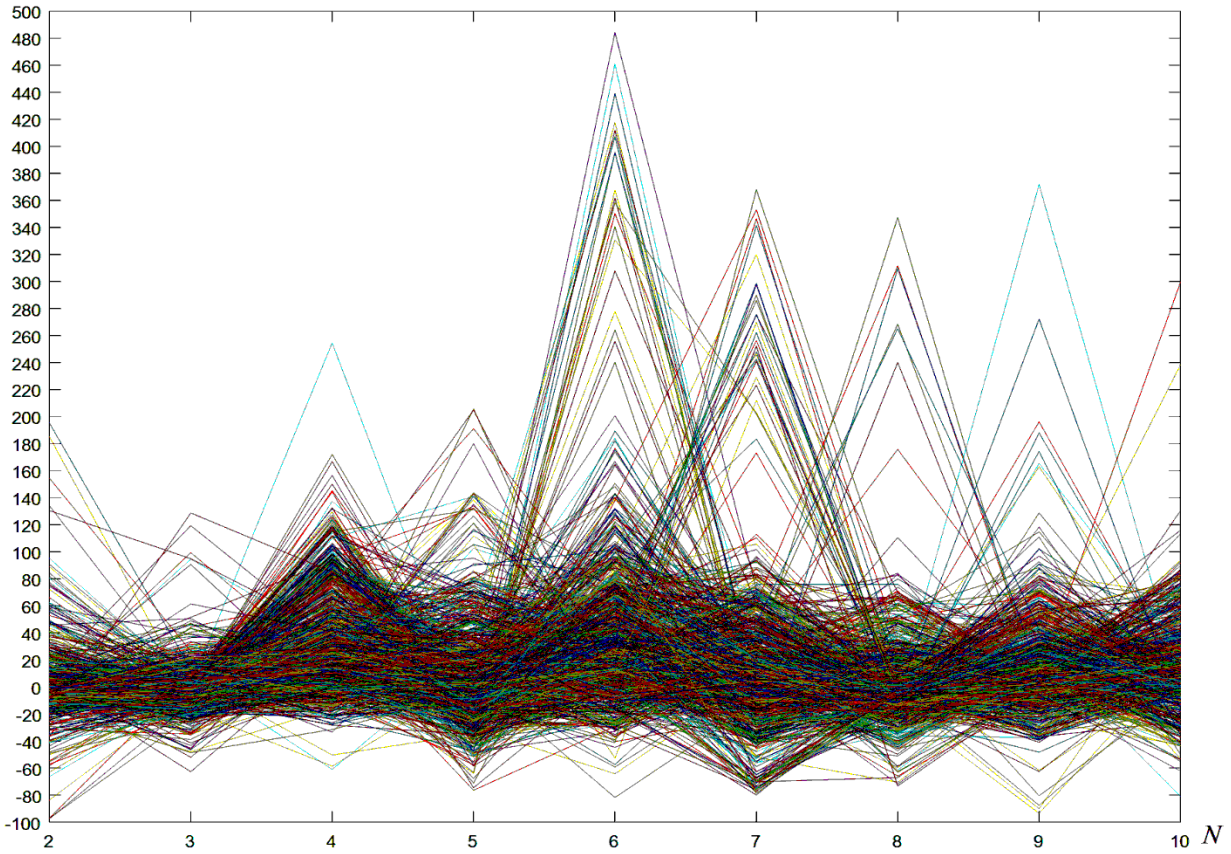


*Fig. 4. Bunches of ratios (19) for 10 cases by (22) — (24) obtained separately in repetitious solutions*

**Discussion**

Even if the plotted polylines suggest that WDJO is computed faster, some of results appear quite contradictive. For instance, considering only polylines in Fig. 1, the inference about the averaged time-advantage of WDJO for the even number of jobs is a bit inconsistent. Polylines in Fig. 3 are more convincing for the case of the even number of jobs, although the case with eight jobs just falls out the mentioned inference. After all, during a single repetition of the problem solving, a negative ratio (19) is not excluded for case (22) — (24), where monotonic PWs are nonetheless formed randomly. So, on average for randomly monotonic PWs, it is better to solve PESP by SLEJIG by the BLPM with WDJO. However, a small possibility of a slower solution (compared to WAJO) still exists.

Cases (20) and (21), which do not contain any random parameters, show us in Fig. 2 that an assured conclusion on WDJO/WAJO advantage must be made for PESPs with six to ten jobs. Indeed, those polylines in Fig. 2 become more “stable” as the number of jobs increases up from 5. Thus, it is faster to solve PESPs with WDJO for 6, 7, 10 jobs in PESP (20), and 6 and 8 jobs in PESP (21). Conversely, PESPs with WAJO for 8 and 9 jobs in PESP (20), and 7, 9, 10 jobs in PESP (21) are solved faster. Note that this time-advantage of WDJO/WAJO here is not averaged. A similar conclusion on 5 and 4 jobs, and especially for 3 and 2 jobs, is not possible. Even by implying “on average”, only the case with 4 jobs is consistent: PESPs (20) and (21) are solved faster by WAJO and WDJO, respectively.

**Conclusion, practical recommendations, and a further research outlook**

WDJO is not definitely better (i. e., faster) than WAJO. For PESPs without randomness with the number of jobs up to six, except for the case of four jobs, a small advantage of WDJO exists only on average. As the number of jobs increases, the advantage of either WDJO or WAJO becomes more certain. For PESPs by SLEJIG which do not have such “regular” PWs, WDJO is expected to be faster than WAJO. However, this averaged time-advantage of WDJO is not truly reliable for all cases (like reliability of just cases with 4, 6, 10 jobs by Fig. 3).

Solving PESPs by SLEJIG with WDJO/WAJO by the BLPM does not have a statistical law of the averaged time-advantage of WDJO or WAJO, which (i. e., the law) could be described in general. To

obtain the gain in computation time, it is recommended to study statistics as per Fig. 1 — 4 before using either WDJO or WAJO. A further research outlook will be focused on WDJO/WAJO in heuristics.

REFERENCES

1. M. L. Pinedo, Scheduling: Theory, Algorithms, and Systems. Springer Int. Publ., 2016.
2. P. Brucker, Scheduling Algorithms. Springer-Verlag Berlin Heidelberg, 2007.
3. H. Belouadah et al., "Scheduling with release dates on a single machine to minimize total weighted completion time", Discrete Applied Mathematics, vol. 36, iss. 3, pp. 213 — 231, 1992.
4. L. P. Fávero and P. Belfiore, "Integer Programming", in: Data Science for Business and Decision Making, Fávero L. P., Belfiore P. (eds.). Academic Press, 2019, pp. 887 — 918.
5. V. V. Romanuke, "Acyclic-and-asymmetric payoff triplet refinement of pure strategy efficient Nash equilibria in trimatrix games by maximinimin and superoptimality", KPI Science News, no. 4, pp. 38 — 53, 2018.

ЛІТЕРАТУРА

1. Pinedo M. L. Scheduling: Theory, Algorithms, and Systems. — Springer Int. Publ., 2016. — 670 p.
2. Brucker P. Scheduling Algorithms. — Springer-Verlag Berlin Heidelberg, 2007. — 371 p.
3. Belouadah H., Posner M. E., Potts C. N. Scheduling with release dates on a single machine to minimize total weighted completion time // Discrete Applied Mathematics. — 1992. — Vol. 36, Iss. 3. — P. 213 — 231.
4. Fávero L. P., Belfiore P. Integer Programming, in: Data Science for Business and Decision Making / Fávero L. P., Belfiore P. (eds.). — Academic Press, 2019. — P. 887 — 918.
5. Romanuke V. V. Acyclic-and-asymmetric payoff triplet refinement of pure strategy efficient Nash equilibria in trimatrix games by maximinimin and superoptimality // KPI Science News. — 2018. — No. 4. — P. 38 — 53.

***Romanuke Vadim Vasylyovych*** *– doctor of technical sciences, professor; Polish Naval Academy, Poland, 81-127, Gdynia, Śmidowicza str., 69; e-mail: romanukevadimv@gmail.com; ORCID: 0000-0003-3543-3087.*

Надійшла – 11.09.2018.