DOI: 10.26565/2304-6201-2019-41-09

UDC 519.161+519.852+519.687.1

A faster way to approximately schedule equally divided jobs with preemptions on a single machine by subsequent job importance growth

V. V. Romanuke
O. S. Popov Odessa National Academy of Telecommunications, Ukraine
romanukevadimv@gmail.com

The goal of this work is to study whether the input order of the job release dates results in different time of computations in finding an approximate schedule for equally divided jobs with preemptions on a single machine by subsequent job importance growth. It has been ascertained that the descending job order has a 1 % relative advantage when scheduling more than 200 jobs. With increasing the number of jobs off 1000, the advantage tends to increase. The advantage can grow up to 22%. A maximally possible gain in computation time is obtained in scheduling longer series of bigger-sized job scheduling problems.

Key words: *heuristic, approximate schedule, job order, preemption, total weighted completion time, computation time gain.*

Мінімізація загального зваженого часу завершення є дуже важливою ціллю в організації та виконанні багатоетапних процесів обчислень, диспетчеризації, виробництва, компонування, будівництва тощо. Ця задача розв'язується теорією розкладів. Теорія розкладів надає підходи для знаходження як точних, так і приблизних розкладів виконання завдань. Точний розклад дозволяє виконати завдання за мінімальний загальний зважений час завершення. Приблизний розклад дозволяє виконати завдання за час, котрий є достатньо близьким до мінімального загального зваженого часу завершення. Однак приблизний розклад отримують незрівнянно швидше, тоді як точні підходи стають практично нездійсненними уже для декількох десятків завдань. Тому розглядається одна ефективна евристика для знаходження приблизного розкладу для випадку однаково розділених завдань з перемиканнями на єдиній машині зі зростанням значущості наступних завдань. При цьому вивчається питання того, чи призводить певний порядок уведення дат запуску завдань до різного часу обчислень. Встановлюється, що спадний порядок завдань має відносну перевагу в 1 % при плануванні більш ніж 200 завдань. Зі зростанням кількості завдань від 1000 ця перевага має незначну тенденцію до зростання. Така перевага може досягти 22 %. Зокрема, послідовність з 240 задач зі 75000 завдань у кожній, де кожне завдання складається з 5 частин, за спадного порядку завдань розплановується за 17.2973 години, тоді як ця ж послідовність за зростаючого порядку завдань розплановується за 21.0691 години. Максимально можливий виграш часу обчислень досягається при роботі з більш довгими послідовностями задач планування завдань більшого розміру.

Ключові слова: *евристика, приблизний розклад, порядок завдань, перемикання, загальний зважений час завершення, виграш у часі обчислення.*

При нахождении приблизительного расписания для одинаково разделённых заданий с переключениями на единственной машине при росте значимости последующих заданий изучается вопрос о том, приводит ли определённый порядок ввода дат запуска заданий к различному времени вычислений. Устанавливается, что убывающий порядок заданий имеет относительное преимущество в 1 % при планировании более чем 200 заданий. С возрастанием количества заданий от 1000 это преимущество имеет незначительную тенденцию к возрастанию. Такое преимущество может достичь 22 %. Максимально возможный выигрыш времени вычислений достигается при работе с более длинными последовательностями задач планирования заданий большего размера.

Ключевые слова: *эвристика, приблизительное расписание, порядок заданий, переключение, общее взвешенное время завершения, выигрыш во времени вычисления.*

Total weighted completion time minimization in the preemptive scheduling problem

In planning, organizing, and executing complex or multistep processes of computing, dispatching, manufacturing, assembling, building, etc., there is a very important goal to minimize the total weighted completion time (TWCT). This problem is addressed by the scheduling theory [1, 2]. The scheduling theory provides approaches to finding both exact and approximate schedules of executing jobs. The exact schedule allows achieving an exactly minimal TWCT. The approximate schedule allows achieving a TWCT which is sufficiently close to the minimum [3]. Nevertheless, the approximate schedule is obtained incomparably faster, whereas the exact approaches become practically intractable just for a few tens of jobs [1].

The schedule is executed either on single machine or multiple machines. Obtaining an optimal single-machine schedule is commonly easier. Minimization of TWCT for a class of preemptive scheduling problems (PESPs), wherein a job can be interrupted in favor of another job [1, 3], is almost thoroughly studied for single-machine problems [4]. Meanwhile, a subclass of single-machine PESPs (SMPESPs) exists, in which importance of subsequent jobs grows, and it can be separated. This is a subclass consisting of SMPESPs by subsequent job importance growth (SJIG).

The subclass of SMPESPs by SJIG

SMPESPs by SJIG refer to those systems whose non-parallelizable development becomes more complicated along with its growing costs. Here, however, the non-parallelizability exists due to unavailability of two or more machines rather than a specific integrity of the system itself. Scheduling by SJIG is a common task in building (or assembling) hierarchical systems/objects whose build-ups above the basis are more complicated and expensive (see, e. g., [1, 2, 5, 6]). SMPESPs for such systems/objects are possible if they "start" functioning only after the assembling completion. A schedule for them is obtained by setting subsequent jobs to greater priority weights. Theoretically, such systems/objects still can be scheduled on multiple machines, so SMPESPs are just a specific case of when only a single machine is available.

An efficient approach to scheduling

In solving SMPESPs, exact schedules are obtainable for no more than a few tens of jobs. In real practice, a lot of heuristics are used to find the approximate schedule. Although the heuristics' approximate schedule is not always executed in the exactly minimal TWCT, the inaccuracy is commonly not so great [1, 2, 4], being often equal to zero. A heuristic known to efficiently give approximate schedules by the minimal inaccuracy is an online scheduling algorithm, which applies the rule of the shortest processing period (SPP) [3].

In solving SMPESPs by SJIG using the SPP approach, there are two ways to input the job release dates and the respective priority weights. On the one hand, the release dates can be given in ascending order. Then the respective priority weights and the release dates will be sets of, generally speaking, non-decreasing values. On the other hand, the release dates can be given in descending order. Then the respective priority weights and the release dates will be sets of, generally speaking, non-increasing values. It is believed that the efficiency of the SPP approach could be increased more by selecting either job order with the shorter computation time (although, surely, both job orders give the same TWCT).

The goal of the article and five stages to achieve it

In finding an approximate schedule for SMPESPs by SJIG using the SPP approach, the goal is to study whether the order of inputting the job release dates results in different time of computations. The significance of the difference, if any, should be shown. As a simplification, the preemptive scheduling of equally divided jobs (EDJs) will be considered. For achieving the said goal for SMPESPs by SJIG of EDJs, the five stages are to be fulfilled:

1. To state an SMPESP by SJIG of EDJs, whose schedule should have no idle time intervals (see, e. g., [1, 2, 4]).
2. To state the SPP approach for finding an approximate schedule.
3. To design a model of generating the SMPESPs by SJIG of EDJs.
4. To estimate the averaged time of obtaining the approximate schedule by both ascending and descending orders of inputting the job release dates.
5. In finding an approximate schedule by the SPP heuristic for SMPESPs by SJIG of EDJs, to discuss and conclude on whether significant the order of inputting the job release dates is.

The SMPESP by SJIG of EDJs

In the SMPESP, the number of jobs is N by $N \in \mathbb{N} \setminus \{1\}$. Job n is divided into H_n equal parts, i. e. job n has a processing period (or time) H_n, which, in the case of EDJs, is actually the same for every job. So, let

$$H_* = H_n \quad \forall\, n = \overline{1,\, N}.$$

It should be noted that the case

$$H_* = 1 \quad \forall\, n = \overline{1,\, N}$$

is excluded from consideration due to the fact that then the SMPESP would be trivial (would have a trivial solution). Job n has a release date r_n and a priority weight w_n, $n = \overline{1,\, N}$. So, in general,

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

are a vector of job processing periods, a vector of job priority weights, and a vector of job release dates, respectively.

Priority weights in vector $\mathbf{W}$ are either non-decreasing for the ascending job order (AJO) or non-increasing for the descending job order (DJO). Formally,

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

for AJO, and

$$w_{l-1}\geqslant w_l\quad\forall l=\overline{2,\,N}\ \text{ but }\ \exists l_*\in\left\{\overline{2,\,N}\right\}\ \text{ such that }\ w_{l_*-1}>w_{l_*} \tag{2}$$

for DJO. Release dates in vector $\mathbf{R}$ are arranged similarly:

$$r_1=1,\ \ r_2=2,\ \ r_{l-1}\leqslant r_l\quad\forall l=\overline{3,\,N}\ \text{ but }\ \exists l_*\in\left\{\overline{3,\,N}\right\}\ \text{ such that }\ r_{l_*-1}<r_{l_*} \tag{3}$$

for AJO, and

$$r_1=N,\ \ r_2=N-1,\ \ r_{l-1}\geqslant r_l\quad\forall l=\overline{3,\,N}\ \text{ but }\ \exists l_*\in\left\{\overline{3,\,N}\right\}\ \text{ such that }\ r_{l_*-1}>r_{l_*} \tag{4}$$

for DJO. Thus, either with (1) by (3) or with (2) by (4), the SMPESP has an SJIG.

The total length of the schedule is $T=\sum\limits_{n=1}^{N}H_*=N\cdot H_*$ (which is measured in definite time units). Job n is completed after moment $\theta\left(n;H_*\right)$, which is

$$\theta\left(n;H_*\right)\in\left\{\overline{1,\,T}\right\}.$$

The goal is to minimize the TWCT, i. e. to schedule those N jobs so that sum $\sum\limits_{n=1}^{N}w_n\theta\left(n;H_*\right)$ would be minimal. The resulting schedule is a set of job tags/numbers $\mathbf{S}=\left[s_t\right]_{1\times T}$ along the grand total of job parts T, where $s_t\in\left\{\overline{1,\,N}\right\}$ for every $t=\overline{1,\,T}$.

The SPP heuristic to solve the SMPESP by SJIG of EDJs

Let $\mathbf{Q}=\left[q_n\right]_{1\times N}=\mathbf{H}$ be a starting vector containing the remaining processing periods (RPPs). Later on, elements of vector $\mathbf{Q}$ will be decreased as time t progresses. Denote an approximate schedule, given by the SPP heuristic, by $\tilde{\mathbf{S}}=\left[\tilde{s}_t\right]_{1\times T}$, where $\tilde{s}_t\in\left\{\overline{1,\,N}\right\}$ for every $t=\overline{1,\,T}$. A set of available jobs

$$A\left(t\right)=\left\{i\in\left\{\overline{1,\,N}\right\}:r_i\leqslant t\text{ and }q_i>0\right\}\subset\left\{\overline{1,\,N}\right\} \tag{5}$$

gives a set of RPP ratios $\left\{w_i/q_i\right\}_{i\in A(t)}$ for every $t=\overline{1,\,T}$, whence the maximal ratio is achieved at subset

$$A^*\left(t\right)=\arg\max_{i\in A(t)}\left\{w_i/q_i\right\}\ \text{ for every }\ t=\overline{1,\,T}. \tag{6}$$

If $\left|A^*\left(t\right)\right|=1$, then

$$\tilde{s}_t=i^*\ \text{ by }\ q_{i^*}^{(\text{obs})}=q_{i^*}\ \text{ and }\ q_{i^*}=q_{i^*}^{(\text{obs})}-1; \tag{7}$$

otherwise, if $\left|A^*\left(t\right)\right|>1$, then a set

$$A^{**}(t) = \arg\max_{i^* \in A^*(t)} w_{i^*} \subset A^*(t) \subset A(t) \tag{8}$$

is found, where

$$A^{**}(t) = \left\{ i_l^{**} \right\}_{l=1}^{L} \subset A^*(t) \subset A(t) \subset \left\{ \overline{1,\, N} \right\}, \tag{9}$$

whence

$$\tilde{s}_t = i_1^{**} \quad \text{by} \quad q_{i_1^{**}}^{(\text{obs})} = q_{i_1^{**}} \quad \text{and} \quad q_{i_1^{**}} = q_{i_1^{**}}^{(\text{obs})} - 1. \tag{10}$$

Then an approximate TWCT is calculated successively for every $n = \overline{1,\, N}$ using the moments

$$\tilde{\theta}(n; H_*) \in \left\{ \overline{1,\, T} \right\}$$

at which each job is completed. Finally,

$$\tilde{\rho}(N) = \sum_{n=1}^{N} w_n \tilde{\theta}(n; H_*) \tag{11}$$

is an approximately minimal TWCT that corresponds to the nearly optimal job schedule $\tilde{\mathbf{S}} = \left[\tilde{s}_t \right]_{1 \times T}$.

A model of generating SMPESPs by SJIG of EDJs

As mentioned above, both the minimal numbers of jobs and job parts are 2. So, let

$$H_* = \overline{2, 14} \quad \forall\, n = \overline{1,\, N} \quad \text{and} \quad N = \overline{2, 1000}. \tag{12}$$

A vector of N priority weights and $N-2$ release dates are generated respectively as follows [7]:

$$w_n = \psi(0.5N\zeta + 1) \quad \forall\, n = \overline{1,\, N} \quad \text{and} \quad r_n = \psi\left((N/3)\zeta + 1\right) \quad \forall\, n = \overline{3,\, N} \tag{13}$$

either with (1) by (3) or with (2) by (4), where ζ is a pseudorandom number drawn from the standard uniform distribution on the open interval $(0; 1)$, and function $\psi(\xi)$ returns the integer part of number ξ. Thus, SMPESPs by SJIG of EDJs will be generated for each H_* and N according to (12): the AJO problems are generated by (13) for (1) and (3); the DJO problems are generated by (13) for (2) and (4). Each problem will be repeated for 100 times to ensure good enough statistical confidence of the results. Mainly, this is about the reliable percentage of a job order relative advantage (PJORA).

Estimation of PJORA and its significance

Denote by $\tau_{As}(k, N)$ and $\tau_{Des}(k, N)$ averaged times of obtaining the heuristic's schedule by AJO and DJO, respectively, for definite H_* and N. The averaging is executed over those 100 repetitions. If

$$\beta(k, N) = 100 \cdot \frac{\tau_{As}(k, N) - \tau_{Des}(k, N)}{\tau_{Des}(k, N)} \tag{14}$$

then AJO has a relative advantage by $\beta(k, N) < 0$, and DJO has a relative advantage by $\beta(k, N) > 0$.

PJORAs (14) for the designed model of generating the SMPESPs by SJIG of EDJs are shown in Fig. 1. As there are a lot of artifacts (unexpected rapid fluctuations of computational speed), including those for a few tens of jobs (for, roughly, $N < 100$), these PJORAs are refreshed in Fig. 2 by ignoring the artifacts. Figure 3 re-refreshes the PJORAs zooming in the range of between 800 and 1000 jobs to schedule.

It is well seen even from Fig. 2 that DJO has a distinct relative advantage when, roughly, $N > 200$. This advantage is about 1 %, which is confirmed by Fig. 3. The overall averaged PJORAs in each of Fig. 1, 2, and 3 prove that DJO is about 1 % faster than ADJ for SMPESPs by SJIG of EDJs. Moreover, the overall averaged PJORA in Fig. 3 prompts that, with increasing the number of jobs off 1000, the advantage has a slight tendency to increase.

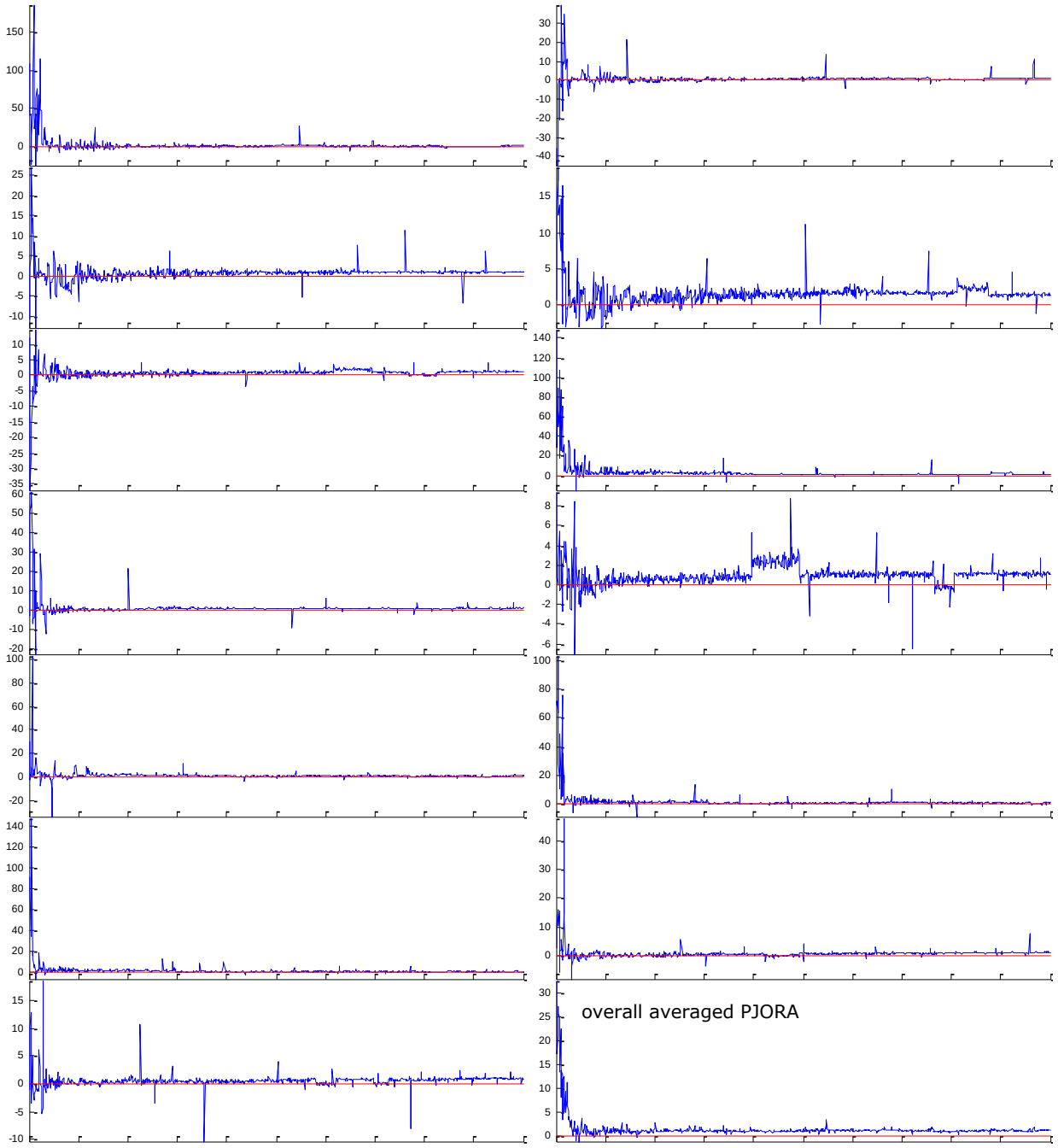

Fig. 1. PJORAs (14) by $H_* = \overline{2,14}$ *(left to right downward) versus* $N = \overline{2,1000}$ *with the horizontal zero level; the overall averaged PJORA (over those 13 PJORAs) is at the right bottom*

Discussion of how significant the job order for SMPESPs by SJIG of EDJs is

At first sight, the 1 % DJO advantage may seem not so significant. Nonetheless, it becomes very significant when scheduling thousands of jobs in a long series of repetitions. For instance, scheduling a series of 80 SMPESPs by SJIG of 25000 jobs divided each into 5 parts takes 2237 seconds with DJO, whereas AJO takes almost 2680 seconds (on the same processing unit). The DJO advantage here is 19.79 %. Another, more impressive example: a series of 240 SMPESPs by SJIG of 75000 jobs divided each into 5 parts is scheduled with DJO in 17.2973 hours, whereas it is scheduled with AJO in 21.0691 hours. The DJO advantage here is 21.81 %. Therefore, the job order significance grows as the size of the SMPESPs is increased. The growth, however, is slow and it seems to be close to an asymptote.

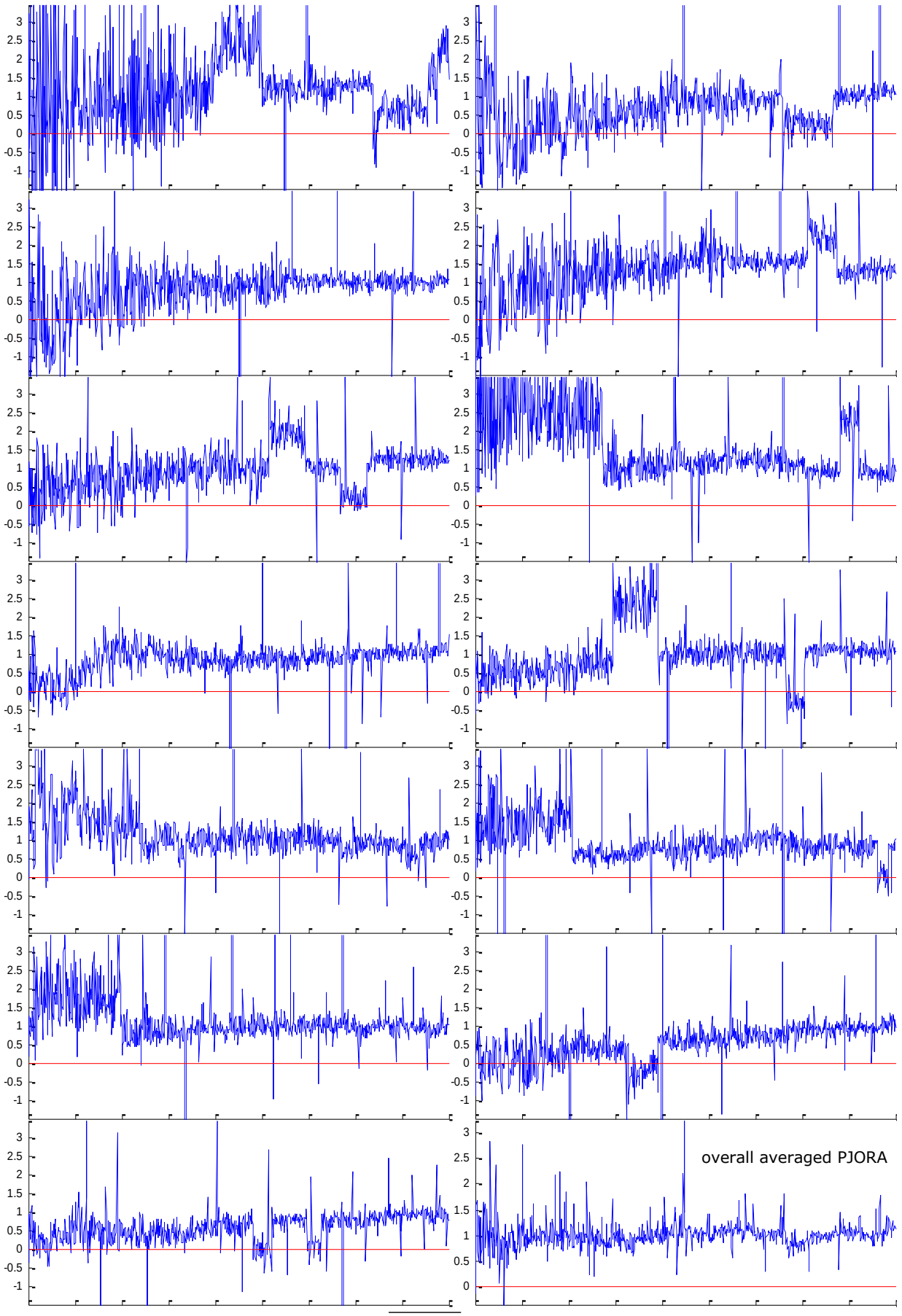

Fig. 2. PJORAs (14) taken off Fig. 1 versus $N = \overline{100,1000}$ *by ignoring the artifacts; the overall averaged PJORA (over those 13 PJORAs) is at the right bottom, wherein no artifacts are cut off*

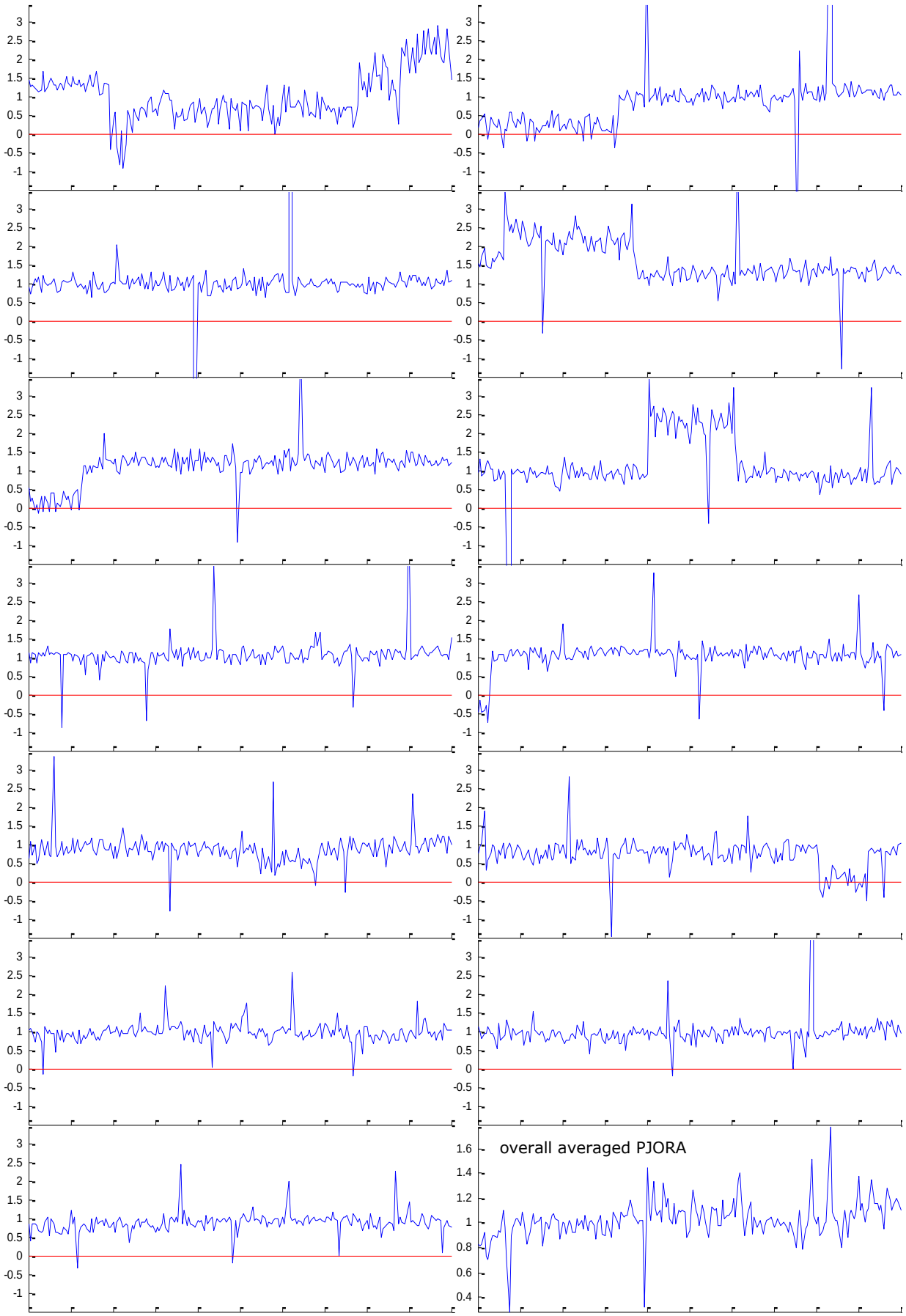

Fig. 3. PJORAs (14) taken off Fig. 1 versus $N = \overline{800,1000}$ *by ignoring the artifacts; the overall averaged PJORA (over those 13 PJORAs) is at the right bottom (the zero level is not seen), wherein no artifacts are cut off*

Conclusion, practical recommendations, and a further research outlook

Based on the well-generalized model, solving SMPESPs by SJIG of EDJs is definitely faster with the DJO input. As the number of jobs increases off a few hundreds, the DJO advantage becomes clearly certain. Nevertheless, in scheduling a fewer jobs (up to a few tens), the AJO input may not concede the DJO input. To get a maximally possible DJO computation time gain, it is recommended to schedule as long series of SMPESPs, as well as SMPESPs of the biggest possible size (including the number of jobs and parts). A further research outlook will be focused on scheduling SMPESPs by SJIG without EDJs.

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. M. L. Pinedo, Planning and Scheduling in Manufacturing and Services. Springer, 2009.
5. I. A. Mandzyuk and V. V. Romanuke, "Rheometric research of polypropylene Licocene PP2602 melts", Archives of Materials Science and Engineering, vol. 50, iss. 1, pp. 31 – 35, 2011.
6. V. V. Romanuke, "Appropriate number and allocation of ReLUs in convolutional neural networks", Research Bulletin of NTUU "Kyiv Polytechnic Institute", no. 1, pp. 69 – 78, 2017.
7. 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. Pinedo M. L. Planning and Scheduling in Manufacturing and Services. Springer, 2009. 536 p.
5. Mandzyuk I. A., Romanuke V. V. Rheometric research of polypropylene Licocene PP2602 melts. *Archives of Materials Science and Engineering*. 2011. Vol. 50, Iss. 1. P. 31 – 35.
6. Romanuke V. V. Appropriate number and allocation of ReLUs in convolutional neural networks. *Research Bulletin of NTUU "Kyiv Polytechnic Institute*. 2017. No. 1. P. 69 – 78.
7. 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; professor of department of information technologies, O. S. Popov Odessa National Academy of Telecommunications, Kuznecznaya str., 1, Odessa, Ukraine, 65029; e-mail: romanukevadimv@gmail.com;*

ORCID: http://orcid.org/0000-0003-3543-3087.

Романюк Вадим Васильович *– доктор технічних наук, професор; професор кафедри інформаційних технологій, Одеська національна академія зв'язку ім. О. С. Попова, вул. Кузнечна, 1, Одеса, Україна, 65029; e-mail: romanukevadimv@gmail.com;*

ORCID: http://orcid.org/0000-0003-3543-3087.

Романюк Вадим Васильевич *– доктор технических наук, профессор; профессор кафедры информационных технологий, Одесская национальная академия связи им. А. С. Попова, ул. Кузнечная, 1, Одесса, Украина, 65029; e-mail: romanukevadimv@gmail.com;*

ORCID: http://orcid.org/0000-0003-3543-3087.

Надійшла – 11.03.2019.