DOI: 10.26565/2304-6201-2019-44-10

UDC 519.161+519.852+519.687.1

Infinity substitute in exactly minimizing total tardiness in tight-tardy progressive 1-machine scheduling by idling-free preemptions of equal-length jobs

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

A schedule ensuring the exactly minimal total tardiness can be found by the respective integer linear programming problem with infinities. In real computations, the infinity which shows that the respective states are either forbidden or impossible is substituted with a sufficiently great positive integer. An open question is whether the substitute can be selected so that the computation time would be decreased. The goal is to ascertain how the increment of the infinity substitute in the respective model influences the computation time of exact schedules. If the influence appears to be significant, then a recommendation on selecting the infinity substitute is to be stated in order to decrease the computation time. A pattern of generating instances of the job scheduling problem is provided. The instances of the job scheduling problem are generated so that schedules which can be obtained trivially, without the exact model, are excluded. Nine versions of the infinity substitute have been proposed. The increment of the infinity substitute in the model of total tardiness exact minimization rendered to solving an integer linear programming problem involving the branch-and-bound approach may have bad influence on the computation time of exact schedules. At least, the greater value of the infinity substitute cannot produce an optimal schedule faster in tight-tardy progressive 1-machine scheduling by idling-free preemptions of equal-length jobs. Roughly the best value of the infinity substitute is the maximal value taken over all the finite triple-indexed weights in the model and increased then by 1. The influence of the “max” infinity substitution is extremely significant. Compared to the case when the infinity is substituted with a sufficiently great integer, the “max” infinity substitution saves up to 50 % of the computation time. This saves hours and even days or months when up to 8 jobs of a few equal processing periods are scheduled for a few thousands of cycles or longer. Therefore, it is strongly recommended always to select the infinity substitute as less as possible in order to decrease the computation time.

Key words: *job scheduling; preemptive 1-machine scheduling; exact model; total tardiness; computation time; infinity substitute.*

Розклад, що забезпечує строго мінімальне загальне запізнювання, можна знайти за відповідною цілочисловою задачею лінійного програмування з нескінченностями. У реальних обчисленнях нескінченність, котра показує, що відповідні стани є забороненими або неможливими, замінюється на достатньо велике додатне ціле. Відкритим є питання про те, чи така заміна може бути підібрана так, щоб час обчислень був би зменшений. Мета полягає у тому, щоб встановити, як збільшення замінника нескінченності у відповідній моделі впливає на час обчислення точних розкладів. Якщо вплив виявиться значним, то слід надати рекомендацію щодо вибору замінника нескінченності для зменшення часу обчислень. Наведено схему генерації екземплярів задачі планування завдань. Екземпляри задачі планування завдань генеруються так, що розклади, які можна отримати тривіально, без точної моделі, виключені. Запропоновано дев’ять варіантів замінника нескінченності. Приріст замінника нескінченності в моделі точної мінімізації загального запізнювання, зведеної до розв’язання цілочислової задачі лінійного програмування, що передбачає підхід методу гілок і меж, може мати поганий вплив на час обчислення точних розкладів. Принаймні більше значення замінника нескінченності не може створити оптимальний розклад швидше у щільному прогресуючому 1-машинному плануванні рівноцінних завдань з перемиканнями без простою. Приблизно найкращим значенням замінника нескінченності є максимум, взятий за усіма скінченними потрійно-індексованими вагами моделі і збільшений потім на 1. Вплив замінника нескінченності “max” є дуже значущим. Порівняно з випадком, коли нескінченність замінена на досить велике ціле, замінник нескінченності “max” заощаджує до 50 % часу обчислень. Це заощаджує години та навіть дні чи місяці, коли розплановується до 8 завдань за кількох рівних періодів до обробки протягом кількох тисяч циклів або довше. Тому настійно рекомендується завжди вибирати замінник нескінченності якомога меншим, щоб скоротити час обчислень.

Ключові слова: *планування завдань; 1-машинне планування з перемиканнями; точна модель; загальне запізнювання; час обчислення; замінник нескінченності.*

Расписание, обеспечивающее строго минимальное общее запаздывание, можно найти по соответствующей целочисленной задаче линейного программирования с бесконечностями. В реальных вычислениях бесконечность, показывающая, что соответствующие состояния запрещены или невозможны, заменяется достаточно большим положительным целым числом. Открытым является вопрос о том, можно ли выбрать замену так, чтобы время вычислений уменьшилось. Цель состоит в том, чтобы выяснить, как приращение заменителя бесконечности в соответствующей модели влияет на время вычисления точных расписаний. Если влияние окажется значительным, то следует дать рекомендацию по выбору заменителя бесконечности для уменьшения времени вычислений. Указана схема генерации экземпляров задачи планирования заданий. Экземпляры задачи планирования заданий генерируются таким образом, что расписания, которые можно получить тривиально, без точной модели, исключаются. Предлагаются девять вариантов замены бесконечности. Приращение заменителя бесконечности в модели точной минимизации общего запаздывания, сведённой к решению целочисленной задачи линейного программирования с использованием подхода ветвления и границ, может плохо влиять на время вычисления точных расписаний. По крайней мере, большее значение заменителя бесконечности не может дать оптимальное расписание быстрее в плотном прогрессирующем 1-машинном планировании равноценных заданий с переключениями без простоя. Примерно лучшим значением заменителя бесконечности является максимум, взятый по всем конечным трёхиндексным весам модели и увеличенный затем на 1. Влияние заменителя бесконечности “max” является очень значительным. По сравнению со случаем, когда бесконечность заменяется достаточно большим целым числом, заменителя бесконечности “max” экономит до 50 %

времени вычислений. Это экономит часы и даже дни или месяцы, когда планируется до 8 заданий с несколькими равными периодами к обработке в течение нескольких тысяч циклов или дольше. Поэтому настоятельно рекомендуется всегда выбирать заменитель бесконечности как можно меньшим, чтобы уменьшить время вычислений.

Ключевые слова: *планирование заданий; 1-машинное планирование с переключениями; точная модель; общее запаздывание; время вычисления; заменитель бесконечности.*

The infinity in exact minimization of total tardiness

The model of exact minimization of total tardiness is rendered to solving an integer linear programming problem involving the branch-and-bound approach [1]. Owing to the fact that no weights are included (see, e.g., [2]), where release dates are set at non-repeating integers from 1 through the total number of equal-length jobs, and due dates are tightly set after the respective release dates (although a few jobs still can be completed without tardiness), the exact model is simplified for such tight-tardy progressive 1-machine scheduling [3]. Theoretically, this model contains infinities which are intended to show that the respective states are either forbidden or impossible. In real computations, the infinity is substituted with a sufficiently great positive integer [3, 4]. An open question is whether the substitute influences the computation time of exact schedules, i.e. whether it is possible to select the infinity substitute so that the computation time would be decreased.

The goal of the research

A basis for the research is the well-known problem of minimizing total tardiness in tight-tardy progressive 1-machine scheduling by idling-free preemptions of equal-length jobs. The goal is to ascertain how the increment of the infinity substitute in the respective model influences the computation time of exact schedules. Occasionally, the influence, if present at all, can be insignificant also. If the influence proves to be significant, then a recommendation on selecting the infinity substitute will be stated in order to decrease the computation time.

The infinity in exactly minimizing total tardiness

The problem of minimizing total tardiness in tight-tardy progressive 1-machine scheduling by idling-free preemptions of equal-length jobs is stated as follows [3, 4]. For N jobs, $N \in \mathbb{N} \setminus \{1\}$, where job n is divided into H equal parts (i.e., has a processing period H), has a release date (the time moment, at which job n becomes available for processing)

$$r_n = n \quad \text{for} \quad n = \overline{1, N} \tag{1}$$

and a due date

$$d_n = H + n - 1 + b_n \quad \text{for} \quad n = \overline{1, N} \tag{2}$$

by a random due date shift

$$b_n = \psi(H \cdot \zeta) \quad \text{for} \quad n = \overline{1, N} \tag{3}$$

with a pseudorandom number ζ drawn from the standard normal distribution (with zero mean and unit variance), and function $\psi(\xi)$ returning the integer part of number ξ (see, e.g., [5]), the goal is to schedule those N jobs so that sum [3, 4, 6]

$$\sum_{n=1}^{N} \max\{0, \theta(n; H) - d_n\} \tag{4}$$

would be minimal, where job n is completed after moment $\theta(n; H)$, which is $\theta(n; H) \in \left\{\overline{1, N \cdot H}\right\}$. This goal is equivalent to minimizing sum

$$\sum_{n=1}^{N} \sum_{h=1}^{H} \sum_{t=1}^{N \cdot H} \lambda_{nht} x_{nht}, \tag{5}$$

where x_{nht} is the decision variable about assigning the h-th part of job n to time moment t: $x_{nht} = 1$ if it is assigned; $x_{nht} = 0$ otherwise. The decision variables are constrained by the following relationships:

$$x_{nht} \in \{0, 1\} \quad \text{by} \quad n = \overline{1, N} \quad \text{and} \quad h = \overline{1, H} \quad \text{and} \quad t = \overline{1, N \cdot H}, \tag{6}$$

$$\sum_{t=1}^{N \cdot H} x_{nht} = 1 \quad \text{by} \quad n = \overline{1, N} \quad \text{and} \quad h = \overline{1, H}, \tag{7}$$

$$\sum_{n=1}^{N} \sum_{h=1}^{H} x_{nht} = 1 \quad \text{by} \quad t = \overline{1, N \cdot H}, \tag{8}$$

$$\sum_{j=t+1}^{N\cdot H}\sum_{h=1}^{H-1}x_{nhj}+Hx_{nHt}\leqslant H \text{ by } n=\overline{1,N} \text{ and } t=\overline{1,N\cdot H-1}. \tag{9}$$

The triple-indexed weights (these ones are not the job priority weights)

$$\left\{\left\{\left\{\lambda_{nht}\right\}_{t=1}^{N\cdot H}\right\}_{h=1}^{H}\right\}_{n=1}^{N} \tag{10}$$

are theoretically defined as follows:

$$\lambda_{nht}=0 \tag{11}$$

by

$$n-1+h\leqslant t\leqslant (N-1)H+h \quad \forall h=\overline{1,H-1} \tag{12}$$

and

$$\lambda_{nht}=\infty \tag{13}$$

when (12) is not true;

$$\lambda_{nHt}=0 \tag{14}$$

by

$$n-1+H\leqslant t\leqslant n-1+H+b_n \tag{15}$$

and

$$\lambda_{nHt}=t-H-n+1-b_n \tag{16}$$

by

$$H+n-1+b_n<t\leqslant N\cdot H \tag{17}$$

and

$$\lambda_{nHt}=\infty \tag{18}$$

when both (15) and (17) are not true. For computations, the infinity in (13) and (18) must be substituted with a sufficiently great positive integer (it can be a sufficiently great positive real number, but the integer is taken for convenience and memory saving in further arithmetic operations). So, instead of (13) and (18),

$$\lambda_{nht}=\alpha \tag{19}$$

and

$$\lambda_{nHt}=\alpha\ , \tag{20}$$

are substituted, respectively, by a sufficiently great positive integer α .

A pattern of generating instances of the job scheduling problem

An optimal job schedule

$$\mathbf{S}^*=\left[s_t^*\right]_{1\times(N\cdot H)} \text{ by } s_t^*\in\left\{\overline{1,N}\right\} \text{ for every } t=\overline{1,N\cdot H} \tag{21}$$

is built by the decision variables at which sum (5) is minimal, where

$$s^*_{\theta^*(n;h)}=n \quad \forall h=\overline{1,H} \text{ by } \theta^*(n;h)\in\left\{\overline{1,N\cdot H}\right\}$$

$$\text{and } \theta^*(n;h)<\theta^*(n;h+1) \text{ for } h=\overline{1,H-1}.$$

Thus, $\theta^*(n;H)$ is a moment after which job n is completed, and, according to sum (4),

$$\vartheta^*(N,H)=\sum_{n=1}^{N}\max\left\{0,\theta^*(n;H)-d_n\right\} \tag{22}$$

is the exactly minimal total tardiness for those N jobs. Meanwhile, schedules for cases with release dates (1) and due dates satisfying inequalities

$$d_n\leqslant d_{n+1} \quad \forall n=\overline{1,N-1} \tag{23}$$

are built trivially, where the job with an earlier release date is scheduled first. To avoid such triviality, due date shifts (3) are re-generated if all inequalities (23) turn to be true. Besides, due date shift (3) for job n is re-generated if $d_n<1$.

At fixed numbers of jobs N and job parts H, for a job scheduling problem instance tagged by an integer c, denote the schedule computation time by $\delta(N,H,c,\alpha)$ by integer α as an infinity substitute in model (1) – (20). Value $\delta(N,H,c,\alpha)$ implies computation time spent on just searching a solution of the respective integer linear programming problem, i.e. on exploring nodes by the branch-and-bound algorithm. If the total number of the instances is C, then the averaged computation time is

$$\tilde{\delta}(N,H,\alpha)=\frac{1}{C}\sum_{c=1}^{C}\delta(N,H,c,\alpha). \tag{24}$$

Computation times (24) will be estimated by $N=\overline{2,8}$ and $H=\overline{2,4}$ for $C=10$ and the following nine versions of the infinity substitute:

1) the infinity is substituted with the sum of all finite triple-indexed weights (10), i.e.

$$\alpha = \sum_{\substack{n=1 \\ \lambda_{nht} \neq \infty}}^{N} \sum_{\substack{h=1 \\ \lambda_{nht} \neq \infty}}^{H} \sum_{\substack{t=1 \\ \lambda_{nht} \neq \infty}}^{N \cdot H} \lambda_{nht} ; \tag{25}$$

2) the infinity is substituted with the maximal value over all finite triple-indexed weights (10) increased by 1, i.e.

$$\alpha = 1 + \max_{\substack{n=\overline{1, N} \\ \lambda_{nht} \neq \infty}} \max_{\substack{h=\overline{1, H} \\ \lambda_{nht} \neq \infty}} \max_{\substack{t=\overline{1, N \cdot H} \\ \lambda_{nht} \neq \infty}} \lambda_{nht} ; \tag{26}$$

3) – 6) the infinity is substituted with multiple maximums in (26) given by

$$\alpha = 2 \cdot \max_{\substack{n=\overline{1, N} \\ \lambda_{nht} \neq \infty}} \max_{\substack{h=\overline{1, H} \\ \lambda_{nht} \neq \infty}} \max_{\substack{t=\overline{1, N \cdot H} \\ \lambda_{nht} \neq \infty}} \lambda_{nht} , \tag{27}$$

$$\alpha = 3 \cdot \max_{\substack{n=\overline{1, N} \\ \lambda_{nht} \neq \infty}} \max_{\substack{h=\overline{1, H} \\ \lambda_{nht} \neq \infty}} \max_{\substack{t=\overline{1, N \cdot H} \\ \lambda_{nht} \neq \infty}} \lambda_{nht} , \tag{28}$$

$$\alpha = 4 \cdot \max_{\substack{n=\overline{1, N} \\ \lambda_{nht} \neq \infty}} \max_{\substack{h=\overline{1, H} \\ \lambda_{nht} \neq \infty}} \max_{\substack{t=\overline{1, N \cdot H} \\ \lambda_{nht} \neq \infty}} \lambda_{nht} , \tag{29}$$

$$\alpha = 5 \cdot \max_{\substack{n=\overline{1, N} \\ \lambda_{nht} \neq \infty}} \max_{\substack{h=\overline{1, H} \\ \lambda_{nht} \neq \infty}} \max_{\substack{t=\overline{1, N \cdot H} \\ \lambda_{nht} \neq \infty}} \lambda_{nht} , \tag{30}$$

respectively;

7) – 9) the infinity is substituted with a sufficiently great integer given by $\alpha = 10^4$, $\alpha = 10^5$, $\alpha = 10^6$.

Obviously, model (1) – (20) cannot be run for infinitely long time. So, its timeout is set at 7200 seconds (i.e., 2 hours). If the solution of the respective integer linear programming problem is not found in 2 hours, the model is stopped and its current solution (which may be non-optimal) is returned.

Computational study

To study how the increment of the infinity substitute in model (1) – (20) influences the computation time of exact schedules, the nine versions of the infinity substitute are to be used. Instead of directly writing integer α into averaged computation time (24), it will be written as a tag within quotation marks by the following correspondence table:

The tag to α	1	2	3	4	5	6	7	8	9
The real value of α	formula (26)	formula (27)	formula (28)	formula (29)	formula (30)	formula (25)	$\alpha = 10^4$	$\alpha = 10^5$	$\alpha = 10^6$

Thus, the averaged computation time is re-denoted as $\tilde{\delta}(N, H, \text{"6"})$ for the case of sum (25), as $\tilde{\delta}(N, H, \text{"1"})$ for maximum (26), as $\tilde{\delta}(N, H, \text{"2"})$ for maximum (27), and so on. Fig. 1 shows averaged computation times (24) versus the nine tags to α (the horizontal axes) as the number of job parts and the job length increase (the entire scheduling problem volume is shown as a stack of small red squares, where the job length is measured horizontally, and the number of job parts is measured vertically). The shortest computation time has occurred 13 times (out of 21 volumes of the entire

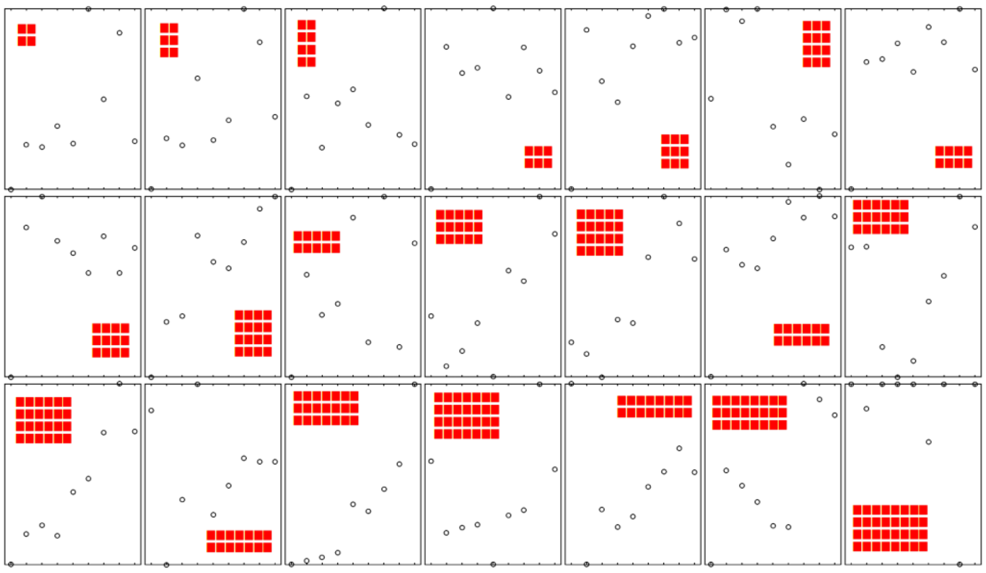

Fig. 1. Averaged computation times (24) versus the nine tags to α (the horizontal axes) by the increasing volume of the entire scheduling problem

scheduling problem) at the infinity substitute as (26) whose value is minimal amongst those nine versions of infinity substitute. The version with the sum by (25) has not resulted in the shortest computation time. Occasionally, $\alpha = 10^5$ has resulted in the shortest computation time twice (at $N = 3$, $H = 4$, and at $N = 8$, $H = 4$), although the case with the greatest volume of the entire scheduling problem ($N = 8$, $H = 4$) has had six timeouts (tags 1, 3, 4, 5, 7, 9). Except for this case, the longest computation time has occurred 14 times at the infinity substitute as a sufficiently great integer given by $\alpha = 10^4$, $\alpha = 10^5$, $\alpha = 10^6$.

The worst cases with the maximal computation times

$$\delta^*(N, H, \alpha) = \max_{c=\overline{1,10}} \delta(N, H, c, \alpha) \tag{31}$$

are shown in Fig. 2 having the 21 stacked subplots analogously to Fig. 1. These subplots have some resemblance to the subplots in Fig. 1. Now, the infinity substitute as $\alpha = 10^5$ (tag 8) has resulted in the shortest computation time only twice, whereas the other versions (tags 1, 2, 3, 4, 5, 6, where integers are far less, indeed) have been the best for the rest of 19 job scheduling problems.

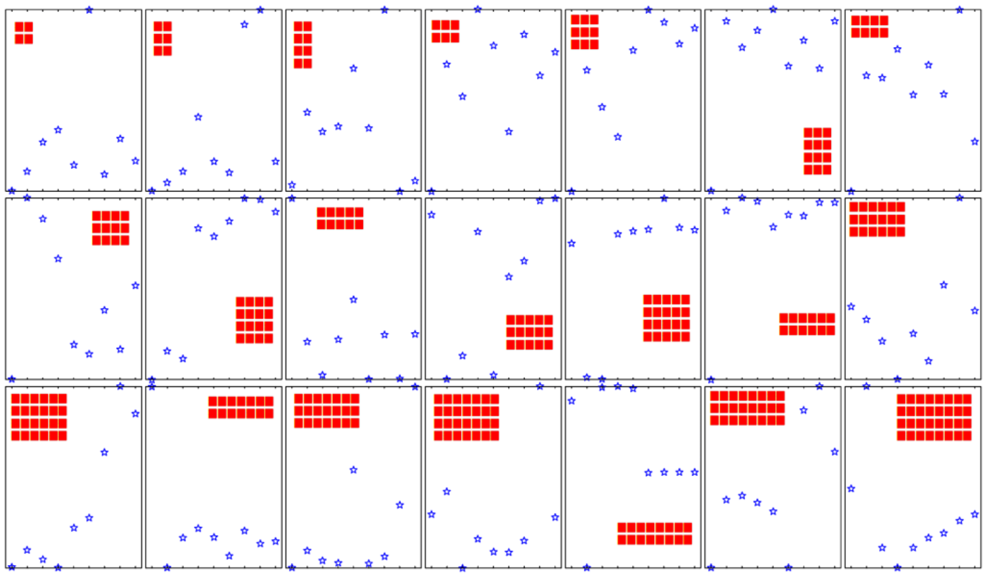

Fig. 2. Maximal computation times (31)

The cases with the minimal computation times

$$\delta_{*}(N, H, \alpha) = \min_{c=\overline{1,10}} \delta(N, H, c, \alpha) \tag{32}$$

are shown in Fig. 3. The resemblance to either Fig. 1 or Fig. 2 is weaker here. The case with the greatest volume of the entire scheduling problem ($N = 8$, $H = 4$) which has had six timeouts (tags 1, 3, 4, 5, 7, 9) and the case with $N = 8$, $H = 2$ are the only two cases where the infinity substitute as $\alpha = 10^5$ (tag 8) has resulted in the shortest computation. Sum (29) has occurred to be the best at the same case with $N = 3$ and $H = 4$. Among the rest of 19 job scheduling problems, the least integer as the infinity substitute (tag 1) by (26) have been the best for 14 cases.

The further averaging is shown in Fig. 4, where each of the three subplots is the average of 21 subplots in Fig. 1 – 3, respectively. The best average computation time (24) has occurred at the least integer as the infinity substitute (tag 1) by (26). It has been close to the best maximal computation time (31), but here sum (25) has eventually “won”. Although the version with $\alpha = 10^5$ (tag 8) has ensured the best minimal computation time (32), it has been an occasional computational artifact caused itself by the six timeouts at the greatest volume of the entire scheduling problem ($N = 8$, $H = 4$).

The first rough inference from the obtained results is that the model with the lesser integer substitute produces an optimal schedule expectedly faster. The difference in real time units (not shown in Fig. 1 – 4, though) depends on the volume of the entire scheduling problem. Nevertheless, the average relative difference between $\tilde{\delta}(N, H, "1")$ and the other versions of the infinity substitute is not less than 2 %. Furthermore, the least value of the infinity substitute produces optimal schedules by 8.51 % to 10.46 % faster than the infinity substitute as a sufficiently great integer given by $\alpha = 10^4$, $\alpha = 10^5$, $\alpha = 10^6$.

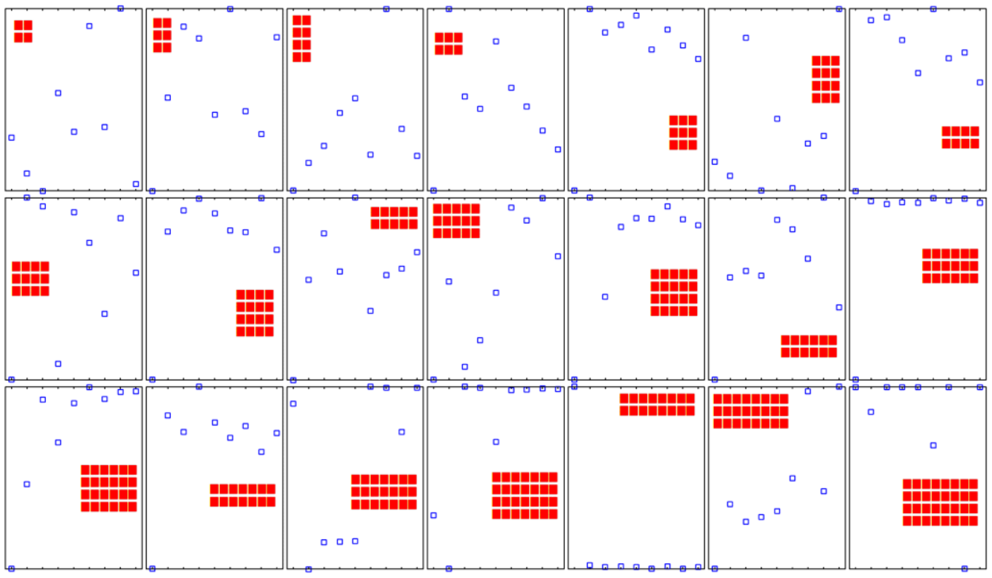
Fig. 3. Minimal computation times (32)

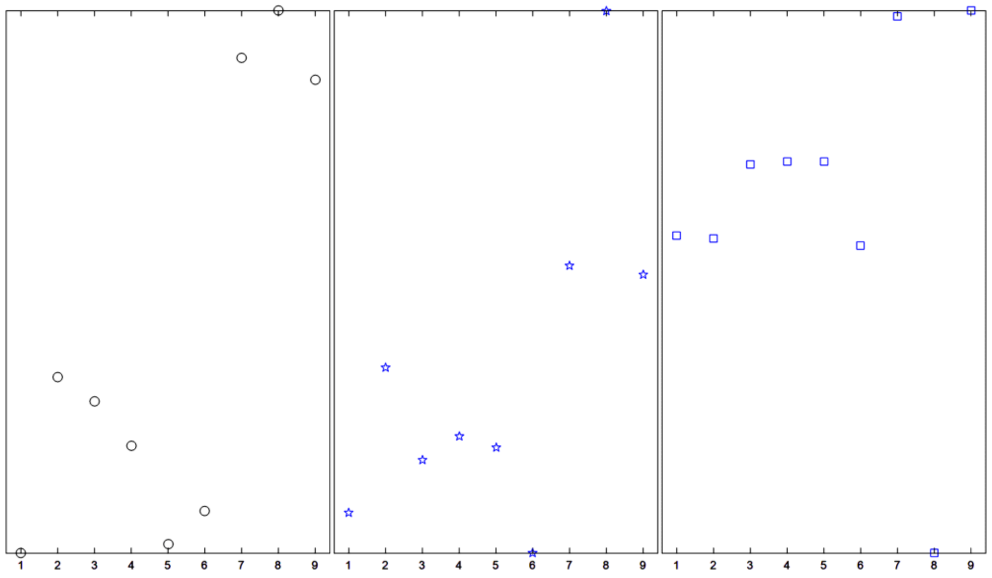

Fig. 4. The averages (from the left to the right) to average computation times (24), to maximal computation times (31), and to minimal computation times (32) versus the nine tags to α *(the horizontal axes)*

Discussion

The main difficulty is that it is hard to find any regularities in the subplots of Fig. 1 – 3. This makes the inference about faster schedules by the lesser integer substitute looser (or rough to some extent), although the inference is expectedly reliable.

The computation time which is saved by substituting the infinity with "max" by (26) instead of using $\alpha = 10^6$ can be really impressive. For example, an optimal schedule

$$\mathbf{S}^* = \left[s_t^* \right]_{1\times 24} = [1 \quad 2 \quad 2 \quad 2 \quad 2 \quad 5 \quad 5 \quad 5 \quad 5 \quad 1 \quad 1 \quad 1 \quad 4 \quad 4 \quad 4 \quad 4 \quad 3 \quad 3 \quad 3 \quad 3 \quad 6 \quad 6 \quad 6 \quad 6] \quad (33)$$

for scheduling 6 jobs by $H = 4$ with due dates

$$\mathbf{D} = \left[d_n \right]_{1\times 6} = [9 \quad 2 \quad 8 \quad 6 \quad 3 \quad 10] \quad (34)$$

is computed in 52.65 seconds by $\alpha = 10^6$, whereas it takes 39.68 seconds to compute schedule (33) with "max" by (26). Total tardiness of schedule (33) is

$$\vartheta^*(6, 4) = \sum_{n=1}^{6} \max\left\{0, \theta^*(n; 4) - d_n\right\} = \max\left\{0, 12 - 9\right\} + \max\left\{0, 5 - 2\right\} +$$

$$+ \max\left\{0, 20 - 8\right\} + \max\left\{0, 16 - 6\right\} + \max\left\{0, 9 - 3\right\} + \max\left\{0, 24 - 10\right\} = 48 .$$

So, the "max" substitution saves here 12.97 seconds (24.64 % of the initial computation time). In another example, for $H = 3$ and due dates

$$\mathbf{D} = \left[d_n \right]_{1\times 8} = \left[4 \quad 4 \quad 5 \quad 7 \quad 5 \quad 8 \quad 10 \quad 13 \right], \tag{35}$$

an optimal schedule

$$\mathbf{S}^* = \left[s_t^* \right]_{1\times 24} = \left[1 \; 1 \; 3 \; 1 \; 3 \; 3 \; 5 \; 5 \; 5 \; 2 \; 2 \; 2 \; 7 \; 7 \; 7 \; 4 \; 4 \; 4 \; 8 \; 8 \; 8 \; 6 \; 6 \; 6 \right] \tag{36}$$

obtained by $\alpha = 10^6$ and an optimal schedule

$$\mathbf{S}^* = \left[s_t^* \right]_{1\times 24} = \left[1 \; 1 \; 1 \; 2 \; 2 \; 2 \; 5 \; 5 \; 5 \; 7 \; 7 \; 7 \; 8 \; 8 \; 8 \; 6 \; 6 \; 6 \; 3 \; 3 \; 3 \; 4 \; 4 \; 4 \right] \tag{37}$$

obtained by the “max” substitution (here the two infinity substitute versions produce slightly different schedules) have the same length as schedule (33), where total tardiness of schedule (36) is

$$\vartheta^*(8, 3) = \sum_{n=1}^{8} \max\left\{0, \theta^*(n; 3) - d_n\right\} = \max\{0, 4-4\} + \max\{0, 12-4\} + \max\{0, 6-5\} +$$
$$+ \max\{0, 18-7\} + \max\{0, 9-5\} + \max\{0, 24-8\} + \max\{0, 15-10\} + \max\{0, 21-13\} = 53$$

being surely the same as total tardiness

$$\vartheta^*(8, 3) = \sum_{n=1}^{8} \max\left\{0, \theta^*(n; 3) - d_n\right\} = \max\{0, 3-4\} + \max\{0, 6-4\} + \max\{0, 21-5\} +$$
$$+ \max\{0, 24-7\} + \max\{0, 9-5\} + \max\{0, 18-8\} + \max\{0, 12-10\} + \max\{0, 15-13\} = 53$$

of schedule (37). However, the difference between the computation times is more impressive here: schedule (36) is computed in 900.736 seconds by $\alpha = 10^6$, whereas it takes 560.61 seconds to compute schedule (37), which is equivalent to (36), with “max”. So, the “max” substitution saves 340.126 seconds (37.76 % of the initial computation time) for the problem with due dates (35). Another, the most demonstrative, example is for scheduling 8 three-parted jobs ($H = 3$) whose due dates are

$$\mathbf{D} = \left[d_n \right]_{1\times 8} = \left[4 \quad 4 \quad 4 \quad 10 \quad 2 \quad 11 \quad 13 \quad 10 \right]. \tag{38}$$

An optimal schedule

$$\mathbf{S}^* = \left[s_t^* \right]_{1\times 24} = \left[1 \; 1 \; 1 \; 3 \; 3 \; 3 \; 5 \; 5 \; 5 \; 8 \; 8 \; 8 \; 4 \; 4 \; 4 \; 6 \; 6 \; 6 \; 7 \; 7 \; 7 \; 2 \; 2 \; 2 \right] \tag{39}$$

obtained by $\alpha = 10^6$ and an optimal schedule

$$\mathbf{S}^* = \left[s_t^* \right]_{1\times 24} = \left[1 \; 1 \; 3 \; 1 \; 3 \; 3 \; 2 \; 2 \; 2 \; 8 \; 8 \; 8 \; 4 \; 4 \; 4 \; 7 \; 7 \; 7 \; 5 \; 5 \; 5 \; 6 \; 6 \; 6 \right] \tag{40}$$

obtained by the “max” substitution (once again the two infinity substitute versions produce slightly different schedules) have the same length as schedules in the previous examples, where total tardiness of schedule (39) is

$$\vartheta^*(8, 3) = \sum_{n=1}^{8} \max\left\{0, \theta^*(n; 3) - d_n\right\} = \max\{0, 3-4\} + \max\{0, 24-4\} + \max\{0, 6-4\} +$$
$$+ \max\{0, 15-10\} + \max\{0, 9-2\} + \max\{0, 18-11\} + \max\{0, 21-13\} + \max\{0, 12-10\} = 51$$

being surely the same as total tardiness

$$\vartheta^*(8, 3) = \sum_{n=1}^{8} \max\left\{0, \theta^*(n; 3) - d_n\right\} = \max\{0, 4-4\} + \max\{0, 9-4\} + \max\{0, 6-4\} +$$
$$+ \max\{0, 15-10\} + \max\{0, 21-2\} + \max\{0, 24-11\} + \max\{0, 18-13\} + \max\{0, 12-10\} = 51$$

of schedule (40). However, the difference between the computation times is very impressive here: schedule (39) is computed in 641.405 seconds by $\alpha = 10^6$, whereas it takes just 326.53 seconds to compute schedule (40), which is equivalent to (39), with “max”. So, for the problem with due dates (38), the “max” substitution saves 314.878 seconds (49.09 % of the initial computation time). And this is just for a single job scheduling problem! Obviously, when such a problem is repeatedly solved, the “max” infinity substitution by (26) saves pretty huge amounts of the computation time. If, for example, there are 10000 job scheduling problems similar to the problem with due dates (38), the “max” substitution saves about 36.4442 days.

Conclusion

The increment of the infinity substitute in the model of total tardiness exact minimization rendered to solving an integer linear programming problem involving the branch-and-bound approach may have bad influence on the computation time of exact schedules. At least, the greater value of the infinity substitute cannot produce an optimal schedule faster in tight-tardy progressive 1-machine scheduling by idling-free preemptions of equal-length jobs. Roughly the best value of the infinity substitute is the maximal value taken over all the finite triple-indexed weights in the model and increased then by 1. Meanwhile, substituting the infinity with just “a sufficiently great positive integer” is not recommended, unless this integer is close to the mentioned maximum or the sum of all the finite triple-indexed weights.

The influence of the “max” infinity substitution is extremely significant. Compared to the case when the infinity is substituted with a sufficiently great integer, the “max” infinity substitution saves up to 50 % of the computation time. This saves hours and even days or months (!) when up to 8 jobs of a few equal parts (processing periods) are scheduled for a few thousands of cycles or longer. Therefore, it is strongly recommended to select the infinity substitute as less as possible in order to decrease the computation time. In scheduling more than 9 jobs with more than 2 processing periods, the exact model becomes practically intractable taking too much computation time whichever infinity substitution is used.

It is uncertain whether those recommendations should be kept for the case when the jobs have different processing periods. So, the research may be furthered by studying this case. Besides, the job priority weights can be also considered for exactly minimizing total weighted tardiness.

REFERENCES

1. M. L. Pinedo, *Scheduling: Theory, Algorithms, and Systems*. Springer Int. Publ., 2016.
2. R. Panneerselvam, “Simple heuristic to minimize total tardiness in a single machine scheduling problem”, *The International Journal of Advanced Manufacturing Technology*, vol. 30, iss. 7 – 8, pp. 722 – 726, 2006.
3. V. V. Romanuke, “Minimal total weighted tardiness in tight-tardy single machine preemptive idling-free scheduling”, *Applied Computer Systems*, vol. 24, no. 2, pp. 150 – 160, 2019.
4. V. V. Romanuke, “Accurate total weighted tardiness minimization in tight-tardy progressive single machine scheduling with preemptions by no idle periods”, *KPI Science News*, no. 5 – 6, pp. 26 – 42, 2019.
5. V. V. Romanuke, “Decision making criteria hybridization for finding optimal decisions’ subset regarding changes of the decision function”, *Journal of Uncertain Systems*, vol. 12, no. 4, pp. 279 – 291, 2018.
6. P. Brucker, *Scheduling Algorithms*. Springer-Verlag Berlin Heidelberg, 2007.

ЛІТЕРАТУРА

1. Pinedo M. L. Scheduling: Theory, Algorithms, and Systems. Springer Int. Publ., 2016. 670 p.
2. Panneerselvam R. Simple heuristic to minimize total tardiness in a single machine scheduling problem. *The International Journal of Advanced Manufacturing Technology*. 2006. Vol. 30, Iss. 7 – 8. P. 722 – 726.
3. Romanuke V. V. Minimal total weighted tardiness in tight-tardy single machine preemptive idling-free scheduling. *Applied Computer Systems*. 2019. Vol. 24, No. 2. P. 150 – 160.
4. Romanuke V. V. Accurate total weighted tardiness minimization in tight-tardy progressive single machine scheduling with preemptions by no idle periods. *KPI Science News*. 2019. No. 5 – 6. P. 26 – 42.
5. Romanuke V. V. Decision making criteria hybridization for finding optimal decisions’ subset regarding changes of the decision function. *Journal of Uncertain Systems*. 2018. Vol. 12, No. 4. P. 279 – 291.
6. Brucker P. Scheduling Algorithms. Springer-Verlag Berlin Heidelberg, 2007. 371 p.

***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 – 11.12.2019.