BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES
Vol. 57, No. 3, 2009
DOI: 10.2478/v10175-010-0122-3

Algorithms for parallel processor scheduling with distinct due windows and unit-time jobs

A. JANIAK[1*], W.A. JANIAK[2], and R. JANUSZKIEWICZ[1]

[1] Institute of Computer Engineering, Control and Robotics, Wrocław University of Technology, 11/17 Janiszewskiego St., 50-372 Wrocław, Poland
[2] Institute of Industrial Engineering and Management, Wrocław University of Technology, 25 Smoluchowskiego St., 50-372 Wrocław, Poland

Abstract. We have studied problems of scheduling n unit-time jobs on m identical parallel processors, in which for each job a distinct due window is given in advance. If a job is completed within its due window, then it incurs no penalty. Otherwise, it incurs a job-dependent earliness or tardiness cost. The objective is to find a job schedule such that the total weighted earliness and tardiness, maximum weighted earliness and tardiness or total weighted number of early and tardy jobs is minimized. Properties of optimal solutions of these problems are established. We proved that optimal solutions for these problems can be found in $O(n^5)$ time in case of minimization of the total weighted earliness and tardiness and the total weighted number of early and tardy jobs and in $O(n^4\sqrt{n \log n})$ time in case of minimization of the maximum weighted earliness and tardiness. The established solution methods are extended to solve the problems with arbitrary integer release dates. A dedicated algorithm with time complexity $O(n^3)$ is provided for the special case of the problem of minimizing total weighted number of early and tardy jobs with agreeable earliness-tardiness weights.

Key words: scheduling algorithms, parallel processor, earliness/tardiness, distinct due windows, unit-time jobs, integer release dates.

1. Introduction

Problems of scheduling unit-time jobs with distinct due windows on identical parallel processors are studied. A due window is a time interval associated with a job. There are no restrictions imposed on due windows. A job incurs no scheduling cost if it is completed within its due window. Otherwise, an earliness or tardiness cost is incurred.

Scheduling problems with distinct due windows model many real-life problems that occur in the production of perishable goods. Consider for example a process in which one chemical is combined with another to produce the final product and one of the chemicals deteriorates rapidly. If the deteriorating chemical is produced before the second chemical is ready it may become useless. On the other hand, if this chemical is produced late, delay in production of the final product may prove costly. Another example comes from a branch of industry, in which a manufacturer agrees with his clients to deliver products in certain time intervals. If the products are manufactured before the earliest acceptable time, they must be kept in warehouses, thus incurring an additional cost. On the contrary, if they are manufactured after the latest acceptable time, the manufacturer must pay for an express delivery of the products.

Scheduling problems with distinct due windows were studied only in a single processor environment. Sidney [1] was among the pioneers, who introduced such problems into scheduling. He studied a special case of a single processor problem with minimization of maximum cost associated with earliness and tardiness, in which any due window is not allowed to contain another due window as a proper subset. He proved that this problem is solvable in polynomial time. However, in general a single processor scheduling problem with distinct due windows and minimization of the total weighted earliness and tardiness is strongly NP-hard, since its special case, i.e. minimization of total weighted tardiness (or earliness), is already strongly NP-hard (see [2]). Some heuristic algorithms for this problem were analyzed in [3] and [4]. Later on in [5] a general case of single processor scheduling problem with distinct due windows and minimization of the weighted number of early and tardy jobs was investigated and the problem was shown to be strongly NP-hard. Some additional discussion on this problem and heuristic algorithms were presented in [6].

The present paper deals with the scheduling problems in which the due windows are given in advance. In a scientific literature there have also been studied problems with due window assignment. For a literature review and some results see [7].

In this paper we analyze problems, whose special cases, i.e. problems of scheduling unit-time jobs with arbitrary job release dates and due dates, have already been studied in the scientific literature. The problems of minimizing the total weighted tardiness and the total weighted number of tardy jobs can be reduced to the network-flow problem (see [8–9]). The problem of minimizing the total weighted number of tardy jobs can also be solved by a polynomial time algorithm constructed by Baptiste et al. [10] for a more general problem with equal processing times. The problem of mini-

*e-mail: adam.janiak@pwr.wroc.pl

mizing the maximum tardiness is solvable by scheduling jobs in the order of non-decreasing due dates [9].

The paper is organized as follows. In Sec. 2, a study of the parallel processor scheduling problem with unit-time jobs and minimization of weighted earliness and tardiness is presented. In Sec. 3, a parallel processor scheduling problem with unit-time jobs and minimization of the weighted number of early and tardy jobs is analyzed. Section 4 concludes the paper.

2. Problem formulation

There are n non-preemptive jobs to be scheduled on m identical parallel processors. Each processor can handle at most one job at a time and each job can be completely processed on any processor. All jobs have unit processing requirements and for each job $j = 1, \ldots, n$ there is given a due window $[\hat{e}_j, \hat{d}_j]$ ($\hat{e}_j \leq \hat{d}_j$), where $\hat{e}_j$ and $\hat{d}_j$ are non-negative integer numbers. A schedule σ determines the allocation of jobs to processors and job starting and completion times. Let $S_j(\sigma)$ and $C_j(\sigma)$ denote the integer start and completion time of job j in schedule σ, respectively. For convenience throughout the paper S_j and C_j are also used instead of $S_j(\sigma)$ and $C_j(\sigma)$, respectively, if there is no possible confusion as to the schedule we refer to. Given a schedule σ, the earliness and tardiness of job j are defined as $E_j(\sigma) = \max\{0, \hat{e}_j - C_j(\sigma)\}$ and $T_j(\sigma) = \max\{0, C_j(\sigma) - \hat{d}_j\}$, respectively. The objective is to find a schedule σ for which one of the following criteria

$$f_{\text{sum}}(\sigma) = \sum_{j=1}^{n} \left(\alpha_j E_j(\sigma) + \beta_j T_j(\sigma)\right),$$

$$f_{\max}(\sigma) = \max_{1 \leq j \leq n} \left\{\alpha_j E_j(\sigma), \beta_j T_j(\sigma)\right\},$$

$$f_{\text{num}}(\sigma) = \sum_{j=1}^{n} \left(\alpha_j V_j(\sigma) + \beta_j U_j(\sigma)\right),$$

is minimized, where

$$V_j(\sigma) = \begin{cases} 1 & \text{if } C_j(\sigma) < \hat{e}_j \\ 0 & \text{in other cases} \end{cases},$$

$$\mathrm{U_j}(\sigma) = \begin{cases} 1 & \text{if } C_j(\sigma) > \hat{d}_j \\ 0 & \text{in other cases} \end{cases},$$

and where α_j and β_j determine strictly positive integer costs of earliness and tardiness, respectively. $V_j(\sigma)$ and $U_j(\sigma)$ are indicators that in the schedule σ job j is early or tardy, respectively. The defined problems will be denoted as P-sum, P-max and P-num, where suffixes "sum", "max" and "num" determine the objective criterion.

3. Properties of optimal solutions

In this section, we provide properties, which allow us to find optimal polynomial time algorithms for the problems formulated in the previous section.

3.1. Minimization of weighted earliness and tardiness.

Lemma 1. There exists an optimal solution for problems *P-sum* and *P-max* for which the starting time S_j of each job j meets the following condition:

$$\max\left\{\hat{e}_j - \left\lceil \frac{n}{m} \right\rceil\right\} \leq S_j \leq \hat{e}_j + \left\lceil \frac{n}{m} \right\rceil - 1.$$

Proof. Assume there is an optimal solution σ, which does not comply with the thesis of this lemma. Therefore, there exists at least one job j, which meets one of the following conditions:

(a) $S_j(\sigma) < \hat{e}_j - \lceil n/m \rceil$,
(b) $S_j(\sigma) > \hat{e}_j + \lceil n/m \rceil - 1$.

Observe that if condition (a) is satisfied for job j then there exists time t such that $\hat{e}_j - \lceil n/m \rceil \leq t \leq \hat{e}_j - 1$ and interval $[t, t+1]$ in which at least one processor is idle. Assume that schedule σ' has been obtained from σ by setting $S_i(\sigma') = S_i(\sigma)$, for $i \neq j$, and $S_j(\sigma') = t$. Then for σ' the objective criterion value is not greater than for solution σ, since $\alpha_j(\hat{e}_j - C_j(\sigma')) < \alpha_j(\hat{e}_j - C_j(\sigma))$ and $\alpha_i(\hat{e}_i - C_i(\sigma')) = \alpha_i(\hat{e}_i - C_i(\sigma))$, for $i \neq j$. Therefore σ' is also optimal. On the other hand, if condition (b) is satisfied for job j then there exists time t such that $\hat{e}_j \leq t \leq \hat{e}_j + \lceil n/m \rceil - 1$ and interval $[t, t+1]$ in which at least one processor is idle. As for the previous case, assume that schedule σ' has been obtained from σ by setting $S_i(\sigma') = S_i(\sigma)$, for $i \neq j$, and $S_j(\sigma') = t$, for which the criterion value is not greater than for the solution σ, since $\beta_j \max\left\{0, C_j(\sigma') - \hat{d}_j\right\} \leq \beta_j \max\left\{0, C_j(\sigma) - \hat{d}_j\right\}$ and $\beta_i \max\left\{0, C_i(\sigma') - \hat{d}_i\right\} = \beta_i \max\left\{0, C_i(\sigma) - \hat{d}_i\right\}$, for $i \neq j$. Therefore σ' is optimal. Repeating the above argument, if necessary, proves the result.

Lemma 2. Let the set of all jobs J be partitioned into two subsets J' and J'', such that $|J'| = k < n$ and $|J''| = n - k$ and $\max_{j \in J'}\{\hat{e}_j\} + \lceil k/m \rceil \leq \min_{j \in J''}\{\hat{e}_j\} - \lceil (n-k)/m \rceil$. There exists an optimal solution to problems P-sum and P-max in which only jobs from J' are scheduled in interval $[\max\{0, \min_{j \in J'}\{\hat{e}_j\} - \lceil k/m \rceil\}, \max_{j \in J'}\{\hat{e}_j\} + \lceil k/m \rceil]$.

Proof. According to Lemma 1

$$\forall_{j \in J'} \left(\max\left\{0, \hat{e}_j - \left\lceil \frac{k}{m} \right\rceil\right\} \leq S_j \leq \hat{e}_j + \left\lceil \frac{k}{m} \right\rceil - 1\right),$$

and

$$\forall_{j \in J''} \left(\max\left\{0, \hat{e}_j - \left\lceil \frac{n-k}{m} \right\rceil\right\} \leq S_j \leq \hat{e}_j + \left\lceil \frac{n-k}{m} \right\rceil - 1\right).$$

Therefore,

$$\forall_{j \in J'} \{S_j, C_j = S_j + 1\} \in$$

$$\in \left[\max\left\{0, \min_{j \in J'}\{\hat{e}_j\} - \left\lceil \frac{k}{m} \right\rceil\right\}, \max_{j \in J'}\{\hat{e}_j\} + \left\lceil \frac{k}{m} \right\rceil\right],$$

and

$$\forall_{j \in J''} \{S_j, C_j = S_j + 1\} \in$$

$$\in \left[\max\left\{0, \min_{j \in J''}\{\hat{e}_j\} - \left\lceil \frac{n-k}{m} \right\rceil\right\}, \max_{j \in J''}\{\hat{e}_j\} + \left\lceil \frac{n-k}{m} \right\rceil\right].$$

Since $\max_{j\in J'}\{\hat{e}_j\} + \lceil k/m\rceil \leq \min_{j\in J''}\{\hat{e}_j\} - \lceil (n-k)/m\rceil$, then the intervals in which the jobs from subsets J' and J" must be scheduled are disjoint.

The following algorithm PARTITION based on Lemma 2 partitions the set of jobs into disjoint subsets. With each subset there is associated an interval in which the jobs from the subset should be scheduled (according to Lemma 1). The intervals associated with different subsets can contain at most one common number – borders of the intervals.

Algorithm PARTITION (input: J – the set of jobs)

1. Let $J_1 = \emptyset$, $J_2 = J$.
2. Let i be the job such that $\hat{e}_i = \min_{j\in J_2}\{\hat{e}_j\}$.
3. Set $J_1 = J_1 \cup \{i\}$, $J_2 = J_2 \setminus \{i\}$.
4. If $J_2 = \emptyset$ then stop.
5. If $\max_{j\in J_1}\{\hat{e}_j\} + \lceil |J_1|/m\rceil \leq \min_{j\in J_2}\{\hat{e}_j\} - \lceil |J_2|/m\rceil$, then J_1 is one of the subsets and to find the other subsets apply PARTITION to J_2,
   else go to step 2.

Lemma 3. If the set of all jobs is partitioned using algorithm PARTITION into $k \leq n$ disjoint subsets $J_1, \ldots, J_k$, then the length of intervals associated with the subsets in which the jobs should be scheduled is at most $O(n\,|J_i|/m)$, for $i = 1, \ldots, k$.

Proof. Suppose the algorithm PARTITION is applied to the set of all jobs. At the beginning there are two subsets, the first one, say J', contains one job and the other one, say J'' contains the rest of jobs. The length of the interval associated with J' equals $2\lceil |J_1|/m\rceil = 2$ and in the worst case

$$\max_{j\in J'}\{\hat{e}_j\} + \left\lceil \frac{|J'|}{m}\right\rceil - 1 = \min_{j\in J''}\{\hat{e}_j\} - \left\lceil \frac{|J''|}{m}\right\rceil.$$

Hence, the distance between the upper bound of the interval associated with J' and $\min_{j\in J''}\{\hat{e}_j\}$ is $\lceil |J''|/m\rceil + 1 \leq \lceil n/m\rceil + 1 = O(n/m)$ and the job with the smallest $\hat{e}_j$ from J'' have to be added to J' and removed from J''. If the described procedure is repeated until $\max_{j\in J'}\{\hat{e}_j\} + \lceil |J'|/m\rceil \leq \min_{j\in J'}\{\hat{e}_j\} - \lceil |J''|/m\rceil$ it is clear that the length of the interval associated with J' is at most $|J'|\,O(n/m) + 2\lceil |J'|/m\rceil = O(n\,|J'|/m)$. If the algorithm PARTITION is next applied to J'' the same reasoning leads to the conclusion that the length of the associated interval is at most $O(n\,|J''|/m)$ and the same for the rest of subsets and the associated intervals.

Lemma 4. If the set J of all jobs is partitioned using algorithm PARTITION into $k \leq n$ disjoint subsets $J_1, \ldots, J_k$, such that the jobs from the subset J_l should be scheduled in interval $[t_{l1}, t_{l2}]$, then an optimal allocation of jobs to the processors and determination of starting times for problems P-sum and P-max can be obtained by solving a proper min-sum or min-max assignment problem for each subset J_l and an associated interval $[t_{l1}, t_{l2}]$.

Proof. According to Lemma 2, it is possible to partition the set J into disjoint subsets, and jobs that belong to each such subset J_l can be scheduled in different interval. Therefore, the objective criterion for problems P-sum and P-max can be rewritten as follows:

$$f_{\text{sum}}(\sigma) = \sum_{j=1}^{n}(\alpha_j E_j(\sigma) + \beta_j T_j(\sigma)) = \sum_{i=1}^{k}\sum_{j\in J_i}(\alpha_j E_j(\sigma) + \beta_j T_j(\sigma))$$

and

$$f_{\max}(\sigma) = \max_{1\leq j\leq n}\{\alpha_j E_j(\sigma), \beta_j T_j(\sigma)\} = \max_{1\leq i\leq k}\left\{\max_{j\in J_i}\{\alpha_j E_j(\sigma), \beta_j T_j(\sigma)\}\right\}$$

respectively, under assumption that $\bigcup_{i=1}^{k} J_i = J$ and $J_i \cap J_j = \emptyset$, for $i, j = 1, \ldots, n$, $i \neq j$. Thus, the minimization of the global objective criterion value can be obtained by the minimization of the criterion value for each disjoint subset of jobs and an associated interval.

For each subset J_l and an associated interval $[t_{l1}, t_{l2}]$ the minimization of the criterion value for the problem P-sum can be formulated as follows:

$$\min : \sum_{i=1}^{i_l}\sum_{j=1}^{j_l} c_{ij}x_{ij}, \tag{1}$$

for $i_l = m(t_{l2} - t_{l1})$ and $j_l = |J_l|$, where

$$c_{(i+q(t_{l2}-t_{l1}))j} = \begin{cases} \alpha_j(\hat{e}_j - (t_{l1}+i)) & \text{if } t_{l1}+i < \hat{e}_j \\ \beta_j\left(t_{l1}+i-\hat{d}_j\right) & \text{if } t_{l1}+i > \hat{d}_j \\ 0 & \text{in other cases} \end{cases} \tag{2}$$

for $1 \leq i \leq t_{l2} - t_{l1}$ and $0 \leq q \leq m-1$, and where j is the j-th job in J_l, subject to

$$\forall_{1\leq i\leq i_l} \sum_{j=1}^{j_l} x_{ij} \leq 1, \tag{3}$$

$$\forall_{1\leq j\leq j_l} \sum_{i=1}^{i_l} x_{ij} = 1. \tag{4}$$

In case of the problem P-max, the minimization of the objective value can be formulated as follows:

$$\min : \max_{1\leq i\leq i_l, 1\leq j\leq j_l}\{c_{ij}x_{ij}\} \tag{5}$$

for $i_l = m(t_{l2} - t_{l1})$ and $j_l = |J_l|$, subject to constraints (3)–(4), where c_{ij} is given by (2). The value x_{ij} should be interpreted as follows:

$$x_{ij} = 1 \Leftrightarrow$$

$$\Leftrightarrow \begin{cases} j\text{-th job in } J_l \text{ is scheduled on processor } \left\lceil \dfrac{i}{t_{l2}-t_{l1}}\right\rceil \\ S_j = t_{l1} + (i-1) \bmod (t_{l2} - t_{l1}) \end{cases} \tag{6}$$

for $1 \leq i \leq i_l$ and $1 \leq j \leq j_l$.

Theorem 1. Optimal solutions for problems P-sum and P-max can be found in $O\left(n^5\right)$ and $O\left(n^4\sqrt{n\log n}\right)$ time, respectively.

Proof. At first the set J of all n jobs is partitioned using algorithm PARTITION into $k \leq n$ disjoint subsets $J_1, \ldots, J_k$. If the jobs in J are sorted according to the non-decreasing order of $\hat{e}_j$ (that takes $O\left(n\log n\right)$ time) then computation in algorithm PARTITION takes $O\left(n\right)$ time. Thus, the operations of sorting and partitioning the jobs take $O\left(n\log n\right)$ time. By Lemma 3, with subset J_i there is associated an interval of length at most $O\left(n|J_i|/m\right)$, for $i = 1, \ldots, k$. The data of the instance of the assignment problem, which by Lemma 4 must be solved for each subset J_i and the associated interval, can be represented as a graph with a number $O\left(n\left|J_i\right|\right)$ of vertices and $O\left(n^2\left|J_i\right|\right)$ of edges. The min-sum assignment problem with a number $|V|$ of vertices and $|E|$ of edges is solvable in $O\left(|V|^2\log|V| + |V|\cdot|E|\right)$ time (see [11]) and the min-max assignment problem is solvable in $O\left(|E|\sqrt{|V|\log|V|}\right)$ time (as a special case of the bottleneck transportation problem with unit edge capacities, see [12]). Therefore, the optimal solution for problem P-sum can be found in

$$\sum_{i=1}^{k} O\left(\left(n\left|J_i\right|\right)^2\log\left(n\left|J_i\right|\right) + n\left|J_i\right|\cdot n^2\left|J_i\right|\right) \leq$$

$$\leq \sum_{i=1}^{k} O\left(\left(n\left|J_i\right|\right)^2\log n^2 + n^3\left|J_i\right|^2\right) =$$

$$= \sum_{i=1}^{k} O\left(\left(n\left|J_i\right|\right)^2\log n + n^3\left|J_i\right|^2\right) =$$

$$= \sum_{i=1}^{k}\left(O\left(n^2\left|J_i\right|^2\log n\right) + O\left(n^3\left|J_i\right|^2\right)\right) =$$

$$= O\left(n^3\right)\sum_{i=1}^{k}\left|J_i\right|^2 \leq O\left(n^5\right)$$

time, and for problem P-max in

$$\sum_{i=1}^{k} O\left(n^2\left|J_i\right|\sqrt{n\left|J_i\right|\log\left(n\left|J_i\right|\right)}\right) \leq$$

$$\leq \sum_{i=1}^{k} O\left(n^2\left|J_i\right|\sqrt{n\left|J_i\right|\log n^2}\right) =$$

$$= O\left(n^2\sqrt{n\log n}\right)\sum_{i=1}^{k}\left|J_i\right|^{1.5} \leq$$

$$\leq O\left(n^2\sqrt{n\log n}\right)\sum_{i=1}^{k}\left|J_i\right|^2 \leq$$

$$\leq O\left(n^4\sqrt{n\log n}\right)$$

time, since $\sum_{i=1}^{k}\left|J_i\right|^2 \leq \left(\sum_{i=1}^{k}\left|J_i\right|\right)^2 = |J|^2 = n^2$.

The solution procedure for problems P-sum and P-max established by Theorem 1 can be extended to the case when for each job j there is given an arbitrary integer release date r_j ($r_j \leq \hat{e}_j$). Notice, that $S_j(\sigma) \geq r_j$ must hold for each job in any feasible schedule σ. If the set of all jobs J is partitioned into disjoint subsets using algorithm PARTITION, then for each subset J_l and job $j \in J_l$ $S_j \in [t_{l1}, t_{l2}]$, where $t_{l1} = \max\left\{0, \min_{i\in J_l}\left\{\hat{e}_i\right\} - \lceil|J_l|/m\rceil\right\}$ and $t_{l2} = \max_{i\in J_l}\left\{\hat{e}_i\right\} + \lceil|J_l|/m\rceil$. Observe that $r_j \leq \max_{i\in J_l}\left\{\hat{e}_i\right\}$ for each $j \in J_l$. Since all the jobs from subset J_l can be scheduled in an even narrower interval $\left[\max_{i\in J_l}\left\{\hat{e}_i\right\}, t_{l2}\right]$, then they can also be scheduled in $[t_{l1}, t_{l2}]$. Thus, it is only necessary to ensure that the starting time of each job is equal to or greater than its release date. To do this, the cost matrix c in (2) has to be replaced with c' defined as follows:

$$c'_{(i+q(t_{l2}-t_{l1}))j} =$$

$$= \begin{cases} M & \text{if } t_{l1}+i < r_j \\ \alpha_j\left(\hat{e}_j - (t_{l1}+i)\right) & \text{if } t_{l1}+i < \hat{e}_j \text{ and } t_{l1}+i \geq r_j \\ \beta_j\left(t_{l1}+i-\hat{d}_j\right) & \text{if } t_{l1}+i > \hat{d}_j \\ 0 & \text{in other cases} \end{cases}, \tag{7}$$

where M is an arbitrary large number, e.g.

$$M = n\sum_{j=1}^{n}\left(\alpha_j + \beta_j\right).$$

Therefore, the following theorem can be stated.

Theorem 2. Optimal solutions for problems P-sum and P-max with arbitrary integer job release dates r_j ($r_j \leq \hat{e}_j$) can be found in $O\left(n^5\right)$ and $O\left(n^4\sqrt{n\log n}\right)$ time, respectively.

3.2. Minimization of weighted number of early and tardy jobs. In the following part of the paper we analyze problems of minimizing the total weighted number of early and tardy jobs. The problem with arbitrary job weights and the problem with agreeable earliness-tardiness weights are studied separately. We provide properties of optimal solutions and optimal polynomial time algorithms to solve the above mentioned problems.

Lemma 5. There exists an optimal solution for problem P-num such that the starting time S_j of any job j meets the following condition:

$$\max\left\{\hat{e}_j - \left\lceil\frac{n}{m}\right\rceil\right\} \leq S_j \leq \hat{e}_j + \left\lceil\frac{n}{m}\right\rceil - 1.$$

Lemma 6. Let the set of all jobs J be partitioned into two subsets J' and J'', such that $|J'| = k < n$ and $|J''| = n-k$ and $\max_{j\in J'}\left\{\hat{e}_j\right\} + \lceil k/m\rceil \leq \min_{j\in J''}\left\{\hat{e}_j\right\} - \lceil(n-k)/m\rceil$. There exists an optimal solution to problem P-num in which only jobs from J' are scheduled in interval $\left[\max\left\{0, \min_{j\in J'}\left\{\hat{e}_j\right\} - \lceil k/m\rceil\right\}, \max_{j\in J'}\left\{\hat{e}_j\right\} + \lceil k/m\rceil\right]$.

Proofs of the above provided lemmas are similar to ones of Lemmas 1 and 2 for problems P-sum and P-max.

Lemma 7. If the set J of all n jobs is partitioned using algorithm PARTITION into $k \leq n$ disjoint subsets $J_1, \ldots, J_k$, such that the jobs from subset J_l should be scheduled in interval $[t_{l1}, t_{l2}]$, then an optimal allocation of jobs to the processors and determination of starting times for problem P-num can be

obtained by solving a proper min-sum assignment problem for each set J_l and an associated interval $[t_{l1}, t_{l2}]$.

Proof. According to Lemma 6, it is possible to partition set J into disjoint subsets, and jobs that belong to each such subset J_l should be scheduled in different interval. Therefore, the objective criterion for problem P-num can be rewritten as follows:

$$f_{\text{num}}(\sigma) = \sum_{j=1}^{n} (\alpha_j V_j(\sigma) + \beta_j U_j(\sigma)) =$$

$$= \sum_{i=1}^{k} \sum_{j \in J_i} (\alpha_j V_j(\sigma) + \beta_j U_j(\sigma))$$

under assumption that $\bigcup_{i=1}^{k} J_i = J$ and $J_i \cap J_j = \emptyset$, for $i, j = 1, \ldots, n$, $i \neq j$.

For each subset J_l and an associated interval $[t_{l1}, t_{l2}]$ the minimization of the criterion value for problem P-num can be formulated as follows:

$$\min : \sum_{i=1}^{i_l} \sum_{j=1}^{j_l} \mathrm{c}_{ij} x_{ij} \tag{8}$$

for $i_l = m(t_{l2} - t_{l1})$ and $j_l = |J_l|$, where

$$c_{(i+q(t_{l2}-t_{l1}))j} = \begin{cases} \alpha_j & \text{if } t_{l1} + i < \hat{e}_j \\ \beta_j & \text{if } t_{l1} + i > \hat{d}_j \\ 0 & \text{in other cases} \end{cases} \tag{9}$$

for $1 \leq i \leq t_{l2} - t_{l1}$ and $0 \leq q \leq m - 1$, and where j is the j-th job in J_l, subject to

$$\forall_{1 \leq i \leq i_l} \quad \sum_{j=1}^{j_l} x_{ij} \leq 1, \tag{10}$$

$$\forall_{1 \leq j \leq j_l} \quad \sum_{i=1}^{i_l} x_{ij} = 1. \tag{11}$$

Value x_{ij} should be interpreted as stated in Eq. (6).

Theorem 3. An optimal solution to problem P-num can be found in time $O(n^5)$.

Proof. Analogous to the proof of Theorem 1.

As for problems P-sum and P-max, the solution method established for problem P-num can be extended to the case when for each job j there is given an arbitrary integer release date r_j ($r_j \leq \hat{e}_j$). To do this, the cost matrix c in (9) has to be replaced with c' defined as follows:

$$c'_{(i+q(t_{l2}-t_{l1}))j} =$$

$$= \begin{cases} M & \text{if } t_{l1} + i < r_j \\ \alpha_j & \text{if } t_{l1} + i < \hat{e}_j \text{ and } t_{l1} + i \geq r_j \\ \beta_j & \text{if } t_{l1} + i > \hat{d}_j \\ 0 & \text{in other cases} \end{cases}, \tag{12}$$

where M is an arbitrary large number, e.g.

$$M = n \sum_{j=1}^{n} (\alpha_j + \beta_j).$$

Thus, the following theorem can be stated.

Theorem 4. An optimal solution to problem P-num with arbitrary integer job release dates r_j ($r_j \leq \hat{e}_j$) can be found in $O(n^5)$ time.

We now pass to consider the special case of problem P-num with agreeable earliness-tardiness weights, i.e. $\alpha_i \geq \alpha_j \Leftrightarrow \beta_i \geq \beta_j$. This simplified problem will be denoted as PA-num. In the following part of the paper there are provided additional properties of problems P-num and PA-num which allow us to construct a dedicated algorithm for problem PA-num.

Lemma 8. For each optimal solution to problem P-num the following properties must be satisfied:

a) if $\alpha_j > \beta_j$ then job j cannot be early;
b) there do not exist two jobs i, j for which $S_i \geq \hat{d}_i$, $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$ and $S_j < \hat{e}_j - 1$ or $S_j \geq \hat{d}_j$;
c) there do not exist two jobs i, j for which $S_i < \hat{e}_i - 1$, $S_j < \hat{e}_j - 1$ and $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$.

Proof. Case (a) Assume that there is an optimal schedule σ in which there exists job j such that $\alpha_j > \beta_j$ and $S_j < \hat{e}_j - 1$. Assume that the schedule σ' has been obtained from σ by setting $S_j(\sigma') = t$ for any $t \geq \hat{d}_j$ such that at least one processor is idle in the interval [t, t+1]. Thus, job j is tardy in schedule σ'. Observe that the objective criterion value for the schedule σ' is smaller than for σ by $\alpha_j - \beta_j$. This contradicts the optimality of the schedule σ.

Case (b) Assume that there is an optimal schedule σ in which there exist two jobs i and j such that $S_i \geq \hat{d}_i$, $S_j < \hat{e}_j - 1$ or $S_j \geq \hat{d}_j$, and $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$. Assume that the schedule σ' has been obtained from σ by setting $S_j(\sigma') = S_i(\sigma)$ and $S_i(\sigma') = t$ for some $t \geq S_i(\sigma)$ (or $t > S_i(\sigma)$ in case of a single processor). Observe that the penalty incurred by jobs i and j in the schedule σ is $\alpha_j + \beta_i$ or $\beta_j + \beta_i$, and in the schedule σ' is at most β_i. Thus, σ cannot be optimal.

Case (c) Assume that there is an optimal schedule σ in which there exist two jobs i and j such that $S_i < \hat{e}_i - 1$, $S_j < \hat{e}_j - 1$ and $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$. Observe that $S_j < S_i$ in σ. Assume that the schedule σ' has been obtained from σ by setting $S_j(\sigma') = S_i(\sigma)$ and $S_i(\sigma') = S_j(\sigma)$. The penalty incurred by jobs i and j in the schedule σ is $\alpha_i + \alpha_j$, and in the schedule σ' is α_i. This contradicts the optimality of σ.

Lemma 9. In each optimal solution to problem P-num if $S_j < \hat{e}_j - 1$ for some job j then for each job i scheduled such that $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$ and $\hat{e}_i - 1 \leq S_i \leq \hat{d}_i - 1$ we have $\alpha_i \geq \alpha_j$.

Proof. Assume there is an optimal schedule σ in which there exist two jobs i and j such that $S_j < \hat{e}_j - 1$, $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$, $\hat{e}_i - 1 \leq S_i \leq \hat{d}_i - 1$ and $\alpha_i < \alpha_j$. Assume that the schedule σ' has been obtained from σ by setting $S_i(\sigma') = S_j(\sigma)$ and $S_j(\sigma') = S_i(\sigma)$. Observe that the penalty incurred by jobs i and j in the schedule σ is α_j, and in the schedule σ' is at most α_i. Since $\alpha_i < \alpha_j$, then σ cannot be optimal.

Lemma 10. In each optimal solution to problem P-num if $S_j \geq \hat{d}_j - 1$ for some job j then for each job i scheduled such that $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$ and $\hat{e}_i - 1 \leq S_i \leq \hat{d}_i - 1$ we have $\beta_i \geq \beta_j$.

Proof. Analogous to the proof of Lemma 9.

Lemma 11. In each optimal solution to problem P-num if $S_i < \hat{e}_i - 1$, $S_j \geq \hat{d}_j$, $S_j \geq \hat{d}_i$, and $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$ for some jobs i and j then $\beta_i \geq \alpha_i + \beta_j$.

Proof. Assume there is given an optimal schedule σ in which exist two jobs i and j such that $S_i < \hat{e}_i - 1$, $S_j \geq \hat{d}_j$, $S_j \geq \hat{d}_i$, $\hat{e}_j - 1 \leq S_i \leq \hat{d}_j - 1$ and $\beta_i < \alpha_i + \beta_j$. Assume that the schedule σ' has been obtained from σ by setting $S_i(\sigma') = S_j(\sigma)$ and $S_j(\sigma') = S_i(\sigma)$. The penalty incurred by jobs i and j in the schedule σ is $\alpha_i + \beta_j$, and in schedule the σ' is β_i. Since $\beta_i < \alpha_i + \beta_j$, then σ cannot be optimal.

On the basis of the above lemmas, we will construct an optimal polynomial time algorithm for the problem PA-num.

Algorithm SCHEDULE (input: J – the set of jobs)

1. Partition J into disjoint subsets using algorithm PARTITION.
2. In each subset J_l sort the jobs in a non-increasing order of $\max\{\alpha_j, \beta_j\}$ and set
   $L_l = \max\{0, \min_{i \in J_l}\{\hat{e}_i\} - \lceil |J_l|/m \rceil\}$ and
   $U_l = \max_{i \in J_l}\{\hat{e}_i\} + \lceil |J_l|/m \rceil$.
3. For each subset J_l until it is empty
   a) Choose from J_l job j with the greatest value of $\max\{\alpha_j, \beta_j\}$ and remove j from J_l.
   b) If there exists an interval $[t, t+1]$ such that $\hat{e}_j - 1 \leq t \leq \min\left\{\hat{d}_j - 1, U_l\right\}$ in which some processor is idle, then set $S_j = t$ and go to a).
   c) If for any interval $[t, t+1]$ there exists job i such that $\hat{e}_i - 1 \leq t \leq \hat{d}_i - 1$ which can be scheduled in a different interval (in which some processor is idle) without introducing an additional penalty, then set $S_j = t$, $j = i$ and go to b) (Lemma 9 and 10, and agreeable earliness-tardiness weights).
   d) If for any interval $[t, t+1]$ such that $\hat{e}_j - 1 \leq t \leq \min\left\{\hat{d}_j - 1, U_l\right\}$ there exists job i which is tardy, then set $S_j = t$, $j = i$ and go to b) (Lemma 8, case b).
   e) If for any interval $[t, t+1]$ such that $\hat{e}_j - 1 \leq t \leq \min\left\{\hat{d}_j - 1, U_l\right\}$ there exists job i which is early and there exists an interval $[t', t'+1]$ such that $L_l \leq t' \leq t$ ($L_l \leq t' < t$ in case of a single processor) in which some processor is idle, then set $S_j = t$, $j = i$ and go to b) (Lemma 8, case c).
   f) If $\alpha_j \leq \beta_j$ and there exists an interval $[t, t+1]$ such that $L_l \leq t < \hat{e}_j - 1$ in which some processor is idle, then set $S_j = t$ and go to a) (Lemma 8, case a).
   g) Choose any interval $[t, t+1]$ such that $\hat{d}_j \leq t \leq U_l$ in which some processor is idle, set $S_j = t$ and go to a).

Theorem 5. Algorithm SCHEDULE solves optimally the problem PA-num in $O\left(n^3\right)$ time.

Proof. Algorithm SCHEDULE is based on the properties established by Lemma 6, 8, 9 and 10, and it relies on the condition that $\alpha_i \geq \alpha_j \Leftrightarrow \beta_i \geq \beta_j$ for any jobs i and j, therefore the solutions obtained by the algorithm are optimal. Assume that the set J was partitioned into $k \leq n$ disjoint subsets $J_1, J_2, \ldots, J_k$. The bound on the time complexity of the algorithm is based on the assumptions that operations in step c) can be done in $O(q)$ time where q is the number of already scheduled jobs and that checking if interval $[t', t'+1]$ exists in step e) can be done in $O(1)$ time. To satisfy the first assumption, for each job there must be stored a number of possible allocations of job to processors within its due window. The number has to be updated for each job whenever some job is scheduled in the interval in which some processor is idle. To fulfill the other assumption, for each

$$t \in \left[\max\left\{0, \min_{i \in J_l}\{\hat{e}_i\} - \lceil |J_l|/m \rceil\right\}, \max_{i \in J_l}\{\hat{e}_i\} + \lceil |J_l|/m \rceil\right]$$

there must be stored a number of idle intervals for all processors within the interval

$$\left[\max\left\{0, \min_{i \in J_l}\{\hat{e}_i\} - \lceil |J_l|/m \rceil\right\}, t\right].$$

The number has to be updated for each t whenever some job is scheduled in the interval in which some processor is idle. Based on the assumptions, the most time demanding operation in step (3) is:

a) finding an interval $[t, t+1]$ which can be done in $O(n|J_l|)$ time, since the length of the interval in which the jobs from subset J_l must be scheduled is $O(n|J_l|/m)$ and there are m processors,
b) updating for each t the number of idle intervals for all processors which can be done in $O(n|J_l|)$ time (since the number for t is easily calculated based on the number for $t-1$).

Assignment of a job to the processor may require a change of the assignment for at most one job. Observe that the change of the assignment for an already scheduled job is possible only in cases c), d) and e) of step (3). If the operations of case c) are performed then the reassignment must be done in case b) of step (3) in the next iteration of the algorithm, hence without further reassignment. On the other hand, if the operations of case d) or e) are performed then the job that must be reassigned is tardy or early, respectively. Thus, the reassignment of this job must be performed is case f) or g) of step (3), hence also without further reassignment. The number of operations required to schedule jobs in the subset J_l can therefore be estimated by $O\left(n|J_l|^2\right)$. The total computation time of algorithm SCHEDULE is

$$\sum_{l=1}^{k} O\left(n|J_l|^2\right) = O(n)\sum_{l=1}^{k}|J_l|^2 \leq$$

$$\leq O(n)\left(\sum_{l=1}^{k}|J_l|\right)^2 = O\left(n^3\right).$$

4. Conclusions

Problems P-sum, P-max and P-num of scheduling n unit-time jobs on m identical parallel processors have been studied, in which with each job is associated a distinct due window and cost of earliness and tardiness. The optimal solutions to problems P-sum and P-num can be found in $O(n^5)$ time and to problem P-max in $O\left(n^4\sqrt{n\log n}\right)$ time by solving a polynomial number of instances of min-sum or min-max assignment problem. The solution methods have been extended to the case of jobs arbitrary integer release dates. For the special case of problem P-num with agreeable earliness-tardiness weights there has been presented a dedicated optimal algorithm with time complexity $O\left(n^3\right)$.

Further research might be focused on development of heuristic solution algorithms for scheduling problems with distinct due windows and arbitrary processing times in the parallel processor environment. These algorithms might be based on the algorithms presented in this paper.

REFERENCES

[1] J. Sidney, "Optimal single-machine scheduling with earliness and tardiness penalties", *Operations Research* 25 (1), 62–69 (1977).

[2] E.L. Lawler, "A pseudopolynomial algorithm for sequence jobs to minimizing total tardiness", *Annals of Discrete Mathematics* 1, 331–342 (1977).

[3] C. Koulamas, "Single-machine scheduling with time windows and earliness/tardiness penalties", *Eur. J. Operational Research* 96, 190–202 (1996).

[4] G. Wan and B.P.-C. Yen, "Tabu search for single machine scheduling with distinct due windows and weighted earliness/tardiness penalties", *Eur. J. Operational Research* 142, 271–281 (2002).

[5] C. Koulamas, "Maximizing the weighted number of on-time jobs in single machine scheduling with time windows", *Mathematical and Computer Modelling* 25 (10), 57–62 (1997).

[6] W.-S. Yoo and L.A. Martin-Vega, "Scheduling single-machine problems for on-time delivery", *Computers & Indusrial Engineering* 39, 371–392 (2001).

[7] A. Janiak, W.A. Janiak, and M. Marek, "Single processor scheduling problems with various models of a due window", *Bull. Pol. Ac.: Tech.* 57 (1), 95–101 (2009).

[8] J. Blazewicz, K.H. Ecker, E. Pesch, G. Schmidt, and J. Weglarz, *Handbook on Scheduling. From Theory to Applications*, Springer, Berlin, 2007.

[9] Peter Brucker, *Scheduling Algorithms*, Springer, Berlin, 2007.

[10] P. Baptiste, P. Brucker, S. Knust, and V. Timkovsky, "Ten notes on equal-execution-time scheduling", *4OR* 2, 111–127 (2004).

[11] M.L. Fredman, R.E. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms", *J. ACM* 34 (3), 596–615 (1987).

[12] A.P. Punnen and R. Zhang, "Bottleneck flows in unit capacity networks", *Information Processing Letters* 109, 334–338 (2009).