Raport Badawczy

Research Report

RB/22/2016

Impact of deadline intervals on behaviour of solutions to the random Sequencing Jobs with Deadlines problem

K. Szkatuła

Instytut Badań Systemowych
Polska Akademia Nauk

Systems Research Institute
Polish Academy of Sciences

POLSKA AKADEMIA NAUK

Instytut Badań Systemowych

ul. Newelska 6

01-447 Warszawa

tel.: (+48) (22) 3810100

fax: (+48) (22) 3810105

Kierownik Zakładu zgłaszający pracę:
Prof. dr hab. inż. Zbigniew Nahorski

Warszawa 2016

Impact of deadline intervals on behaviour of solutions to the random Sequencing Jobs with Deadlines problem

Krzysztof SZKATUŁA
Systems Research Institute, Polish Academy of Sciences
ul. Newelska 6, 01-447 Warszawa, Poland
Siedlce University of Natural Sciences and Humanities
ul. Konarskiego 2, 08-110 Siedlce, Poland
E-mail: Krzysztof.Szkatula@ibspan.waw.pl

October 18, 2016

Abstract

The paper analyzes the influence, exerted by the mutual relations of deadline intervals on behaviour of the optimal solution values for the random Sequencing Jobs with Deadlines (SJD) problems. An asymptotically sub-optimal algorithm is proposed. It is assumed that the problem coefficients are realizations of independent uniformly distributed random variables and deadlines are deterministic. The results, presented in the paper, significantly extend knowledge on behaviour of the optimal solutions to the SJD problem in the asymptotical case.

Keywords: Scheduling, Combinatorial optimization, Probabilistic Analysis, Approximate Algorithm, Profit

1 Introduction

The *sequencing jobs with deadlines* problem (SJD) consists in maximizing the weighted number of jobs processed before their deadlines. Deadlines may be considered as special cases of due windows (due intervals), see Janiak et al. [8]. Each job j $(j = 1, ..., n)$ is to be processed on a single machine. It requires a processing time t_j and has a deadline $d_j(n)$. Deadlines are assumed to be the functions of n in order to allow for the asymptotical analysis of SJD, when $n \to \infty$. If the job is completed before its deadline, a profit p_j is earned. The objective is to maximize the total profit, which could be considered as equivalent to minimizing the total cost or minimizing the weighted sum of late jobs.

From the point of view of the deterministic scheduling problems theory, the SJD problem belongs to the class of the single machine scheduling (SMS) problems. More precisely, it is considered as a scheduling problem with optimisation criteria involving due dates, classified, according to Graham notation, as $1 || \Sigma w_j U_j$, see Błażewicz et al. [3], p. 106. There are many research papers that deal with SMS problems, both due to their own scientific value and as parts of more generalized and complex problems. The SJD problem often occurs as a sub-problem in various sequencing and scheduling problems. In Baptiste et al. [1] job sequencing problems with due dates and deadlines were considered. Paper by Catanzaro et al. [4] addressed the job sequencing problems with tool switching. In Baptiste and Le Pape [2] scheduling problems with setup constraints were analyzed. Detienne [5] considered scheduling problems with machine availability constraints. These ones are only few, out of many, problems where the SJD problem is included as the sub-problem. In many cases, the SJD problem may be used as relaxation of the more complex problems.

It is assumed, with insignificant loss of generality, that jobs are indexed according to

$$d_1(n) \leq d_2(n) \leq \cdots \leq d_n(n). \tag{1}$$

Then, the SJD problem can be formulated as a binary (0-1) programming problem (cf. Lawler and Moore [11]):

$$\begin{aligned} & z_{OPT}(n) = \max \sum_{j=1}^{n} p_j x_j \\ \text{s.t.} \quad & \sum_{j=1}^{i} t_j x_j \leq d_i(n), \quad i = 1, \ldots, n \\ & \text{where} \quad x_j = 0 \text{ or } 1, \quad j = 1, \ldots, n \end{aligned} \tag{2}$$

where $x_j = 1$ only if job j is completed before its deadline. Jobs on time should be processed in the order conform to (1), while completion of the late jobs is of no importance, since no profit is earned. If all $p_j = 1$, $j = 1, \ldots, n$, then the optimization goal is to maximize the number of jobs performed within the deadlines or, equivalently, to minimize the number of tardy jobs. Without loss of generality we also assume that

$$0 < t_j \leq d_j(n) \text{ and } p_j > 0,\ j = 1, \ldots, n.$$

SJD is well known to be an NP-hard problem, see Garey and Johnson [7], but it can be solved in a pseudopolynomial time by a dynamical programming method of Sahni [14]. In the literature, many algorithms have been proposed to solve the sequencing or scheduling jobs on a single machine. Many of the proposed solution techniques are based on (mixed) integer linear programming problem formulations, cf. Baptiste et al. [1], Catanzaro et al. [4] and Detienne [5]. Another general technique which could be used is Branch and Bound method, see Baptiste and Le Pape [2]. There were also attempts to use genetic algorithms, see Sevaux and Dauzere-Péres [15] or the neural network approach, cf. El-Bouri et al. [6]. In the paper by Levner and Elalouf, see [12], an improved version of the polynomial-time approximation algorithm to solve the SJD problem was presented. The above list of references has illustrative purpose and it is far from being exhaustive.

In the literature, a certain simplified version of the SJD problem was considered. In this case all jobs have identical processing times, i.e. $t_i = c$, $c > 0$, $i = 1, \ldots, n$ where c is some constant. For this version of the SJD problem many efficient greedy type algorithms were proposed, cf. Puntambekar [13]. Moreover, greedy type algorithms are often used in this context in the teaching process at the universities, cf. Kocur [10].

It can be easily observed that SJD is a special case of the well known binary (0-1) multi-constraint knapsack problem, cf. Kellner et al. [9], according to the following formulation:

$$z_{OPT}(n) = \max \sum_{i=1}^{n} c_i x_i \text{ s.t.} \sum_{i=1}^{n} a_{ji} x_i \leq b_j(n),\ x_i \in \{0,1\},\ i,\ j = 1, \ldots, n \tag{3}$$

where, in (3), $c_j = p_j$, $a_{ij} = t_j$, $1 \leq i \leq j$, $a_{ij} = 0$, $j < i \leq n$, $b_j(n) = d_j(n)$, $j = 1, \ldots, n$. When all constraints, but last, in (2) are dropped, then SJD problem is reduced to the classical (single constraint) knapsack problem:

$$z_{OPT}(n) = \max \sum_{i=1}^{n} p_i x_i \text{ s.t.} \sum_{i=1}^{n} t_i x_i \leq d_n(n),\ x_i \in \{0,1\},\ i = 1, \ldots, n. \tag{4}$$

It is well known that multi-constraint knapsack problem is NP hard in the strong sense, while both SJD and single-constraint knapsack problems are NP hard but not in the strong sense, cf. Garey and Johnson [7].

There are various approaches to deal with uncertainty, e.g. defined as randomness of the problem data (coefficients), in the case of job sequencing or scheduling problems, cf. Xia et al. [17]. In the paper by Szkatuła [16] asymptotic growth (as $n \to \infty$) of the value of $z_{OPT}(n)$ for the class of random SJD problems was analyzed. The goal of the present paper is to investigate the influence of the intervals of deadlines on asymptotical behaviour (as $n \to \infty$) of the optimal solution values $z_{OPT}(n)$ in the case of random version of the SJD problem, where intervals of deadlines are defined by behaviour of $d_1(n), d_2(n) - d_1(n), \ldots, d_n(n) - d_{n-1}(n)$,more precisely by their mutual relations. A simple heuristic algorithm for solving the SJD problems is proposed and it is proven that in the average case it is asymptotically sub-optimal. The obtained results are significantly extending the ones presented in the paper by Szkatuła [16].

The results achieved constitute a contribution to the field of scheduling problems as well as to the probabilistic analysis of the combinatorial optimization problems. These results could be also useful for constructing and testing approximate algorithms for solving SJD problems.

The following notation is used throughout the paper: $V_n \approx Y_n$, $n \to \infty$ denotes:

- $Y_n \cdot (1 - o_n(1)) \leq V_n \leq Y_n \cdot (1 + o_n(1))$ if V_n and Y_n are sequences of numbers;
- $\lim_{n\to\infty} P\{Y_n \cdot (1 - o_n(1)) \leq V_n \leq Y_n \cdot (1 + o_n(1))\} = 1$ if V_n is a sequence of random variables and Y_n is a sequence of numbers or random variables, where $o_n(1) > 0$ and $\lim_{n\to\infty} o_n(1) = 0$, as usual.

In Section 2 some useful duality estimations of (2) are presented. These estimations are exploited in Section 3, which presents probabilistic analysis of the SJD problem. Both Sections 2 and 3 are partially based on the paper by Szkatuła [16]. Please refer to it for a more detailed presentation. Section 4 contains the main results of the paper, related to the deadline intervals and the approximate algorithm. Section 5 discusses obtained results.

2 Lagrange function and dual estimations

Let us consider the Lagrange function of (2), cf. Szkatuła [16]:

$$\begin{aligned} F_n(x,\Lambda) &= \sum_{j=1}^{n} p_j x_j + \sum_{i=1}^{n} \lambda_i \cdot \left(d_i(n) - \sum\nolimits_{j=1}^{i} t_j x_j \right) = \\ &= \sum_{i=1}^{n} \lambda_i d_i(n) + \sum_{j=1}^{n} (p_j - \Lambda_j \cdot t_j) \cdot x_j \end{aligned}$$

where $x = \{x_1, \ldots, x_n\}$, $\Lambda = \{\lambda_1, \ldots \lambda_n\}$, $\Lambda_j = \sum_{i=j}^{n} \lambda_i$. Let for every Λ, $\lambda_j \geq 0,\ j = 1, \ldots, n$

$$\begin{aligned} \varphi_n(\Lambda) &= \max_{x \in \{0,1\}^n} F_n(x,\Lambda) = \sum_{i=1}^{n} \lambda_i d_i(n) + \sum_{j=1}^{n} (p_j - \Lambda_j \cdot t_j) \cdot x_j(\Lambda_j) = \\ &= \sum_{j=1}^{n} p_j(\Lambda_j) + \sum_{i=1}^{n} \lambda_i \cdot \left(d_i(n) - \sum\nolimits_{j=1}^{i} t_j(\Lambda_j) \right) \end{aligned}$$

where

$$\begin{array}{llllll} \text{if} & p_j - \Lambda_j t_j > 0 & \text{then} & x_j(\Lambda_j) = 1; & p_j(\Lambda_j) = p_j; & t_j(\Lambda_j) = t_j; \\ \text{if} & p_j - \Lambda_j t_j \leq 0 & \text{then} & x_j(\Lambda_j) = 0; & p_j(\Lambda_j) = 0; & t_j(\Lambda_j) = 0. \end{array} \tag{5}$$

Let us denote:

$$\begin{aligned} z_n(\Lambda) &= \sum_{j=1}^{n} p_j(\Lambda_j);\ s_i(\Lambda) = \sum_{j=1}^{i} t_j(\Lambda_j); \\ \hat{p}(\Lambda_j) &= \begin{cases} p_j(\Lambda_j) & \text{if } s_j(\Lambda) \leq d_j(n) \\ 0 & \text{otherwise} \end{cases};\ \hat{z}_n(\Lambda) = \sum_{j=1}^{n} \hat{p}(\Lambda_j); \\ D_n(\Lambda) &= \sum_{i=1}^{n} \lambda_i \cdot d_i(n);\ S_n(\Lambda) = \sum_{i=1}^{n} \lambda_i \cdot s_i(\Lambda) = \\ &= \sum_{j=1}^{n} \Lambda_j \cdot t_j(\Lambda_j);\ \varphi_n(\Lambda) = z_n(\Lambda) + D_n(\Lambda) - S_n(\Lambda). \end{aligned}$$

The problem dual to SJD (2) is then as follows:

$$\Phi_n^* = \min_{\Lambda \geq 0} \varphi_n(\Lambda).$$

By the construction of $z_n(\Lambda)$, $\hat{z}_n\ (\Lambda)$, $S_n(\Lambda)$, $D_n(\Lambda)$, $\varphi_n(\Lambda)$ and $\Phi_n^*(\Lambda)$ we have for any $\Lambda \geq 0$:

$$z_n(\Lambda) \geq S_n(\Lambda)$$

and

$$\hat{z}_n(\Lambda) \leq z_{OPT}(n) \leq \Phi_n^* \leq \varphi_n(\Lambda) = z_n(\Lambda) + D_n(\Lambda) - S_n(\Lambda). \tag{6}$$

3 Probabilistic analysis

The following random model of the SJD problem (2), cf. Szkatuła [16], will be considered:

- $n \geq 1$, n is a positive integer, $n \to \infty$, $i,\ j = 1, \ldots, n$
- $t_j,\ p_j,$ are realizations of mutually independent random variables and moreover $t_j,\ p_j$, are uniformly distributed over the $(0, 1]$ interval;
- $0 \leq d_1(n) \leq d_2(n) \leq \cdots \leq d_n(n)$ and $\Lambda = \{\lambda_1, \ldots, \lambda_n\}$ are deterministic, $d_j(n)$ are functions of n.

Let us compute the distributions and average values of $t_i(\Lambda_i)$, $p_i(\Lambda_i)$, as functions of n, Λ_i in the asymptotical case, when $n \to \infty$:

$$\begin{aligned} G_i(\Lambda_i, x) &= P\{t_i(\Lambda_i) < x\} = P\{t_i < x \cup t_i > x \cap p_i < \Lambda_i \cdot t_i\} = \\ &= \begin{array}{lll} \left.\begin{array}{ll} \frac{\Lambda_i}{2} + x\left(1 - \frac{x \cdot \Lambda_i}{2}\right) & \Lambda_i \leq 1 \\ 1 - \frac{1}{2\Lambda_i} + x \cdot \left(1 - \frac{x \cdot \Lambda_i}{2}\right) & \Lambda_i > 1 \end{array}\right\} & \text{if} & 0 < x \leq \min\left\{1, \frac{1}{\Lambda_i}\right\} \\ 1 & \text{if} & x > \min\left\{1, \frac{1}{\Lambda_i}\right\} \end{array} \end{aligned}$$

$$\begin{aligned} H_i(\Lambda_i, x) &= P\{p_i(\Lambda_i) < x\} = P\{p_i < x \cup t_i > x \cap p_i < \Lambda_i \cdot t_i\} = \\ &= \begin{cases} 1 - \frac{1 - x^2}{2 \cdot \Lambda_i} & \text{if } \Lambda_i \geq 1 \\ \frac{\Lambda_i}{2} + \frac{x^2}{2 \cdot \Lambda_i} & \text{if } x \leq \Lambda_i \leq 1 \\ x & \text{if } \Lambda_i \leq x \leq 1 \\ 1 & \text{if } x \geq 1 \end{cases} \end{aligned}$$

$$E(t_i(\Lambda_i)) = \int_0^1 x \cdot dG_i(\Lambda_i, x) = \begin{cases} \frac{1}{6 \cdot \Lambda_i^2} & \text{if } \quad \Lambda_i \geq 1 \\ \frac{1}{2} - \frac{\Lambda_i}{3} & \text{if } \quad 0 \leq \Lambda_i \leq 1 \end{cases}$$

$$E(p_i(\Lambda_i)) = \int_0^1 x \cdot dH_i(\Lambda_i, x) = \begin{cases} \frac{1}{3 \cdot \Lambda_i} & \text{if } \quad \Lambda_i \geq 1 \\ \frac{1}{2} - \frac{\Lambda_i^2}{6} & \text{if } \quad 0 \leq \Lambda_i \leq 1. \end{cases}$$

The asymptotical analysis of the SJD problem consists in observing the behaviour of certain problem characteristics when $n \to \infty$. Therefore, it is assumed that some of the problem parameters may functionally depend on n, which is denoted by (n). This especially applies to deadlines $d_i(n)$ and Lagrange multipliers $\Lambda(n)$, $\lambda_i(n)$, $\Lambda_i(n)$, $i = 1, \ldots, n$. This notation will be used when dependence on n is essential in terms of the context, and may be omitted, for brevity purposes, in general formulas, when dependence on n is not important.

We are looking for such $\lambda_1(n), \ldots, \lambda_n(n) \geq 0$ that

$$E(s_i(\Lambda_i(n))) \leq d_i(n), \text{ for all } i = 1, \ldots, n. \tag{7}$$

By assumption $\Lambda_j = \sum_{i=j}^{n} \lambda_i$, $\lambda_j \geq 0$, $j = 1, \ldots, n$, and then we have

$$\Lambda_1 \geq \Lambda_2 \geq \cdots \geq \Lambda_n \text{ and therefore } E(t_1(\Lambda_1)) \leq E(t_2(\Lambda_2)) \leq \cdots \leq E(t_n(\Lambda_n)). \tag{8}$$

Let us observe that if for certain i, $1 \leq i < n$

$$E(s_i(\Lambda(n))) = d_i(n) \text{ and } d_{i+1}(n) - d_i(n) < E(t_{i+1}(\Lambda_{i+1}(n))) \qquad (9)$$

then

$$E(s_{i+1}(\Lambda(n)) > d_{i+1}(n)$$

which means that if for some sets of deadlines $d_1(n), d_2(n), \cdots, d_n(n)$ the vector $\Lambda(n)$, such that (9) holds, is determined, then (7) will not be fulfilled for all $i = 1, \ldots, n$. This will be caused by monotonicity of Λ_j(n) and $E(t_j(\Lambda_j(n))$, see (8). Hence, in Szkatuła [16] a recursive Algorithm 1 determining $\Lambda(n)$, $\lambda_i(n) \geq 0$, $i = 1, \ldots, n$, guaranteeing that for each $d_1(n) \leq d_2(n) \leq \cdots \leq d_n(n)$ (7) will be fulfilled, was proposed. This algorithm is presented below in a new and simpler formulation, which is, moreover, more suitable for further presentation.

Algorithm 1 ***Procedure to determine $\delta_j(n)$ and $\Lambda_j(n)$, $j = 1, \ldots, n$***

Initialization Step: Let $l \leftarrow 0$, $d_0(n) \leftarrow 0$
Main Recursive Step: Let

$$j^* = \max_{l<j\leq n} \left\{ j \;\middle|\; d_{j-1}(n) \geq \frac{j-1}{j} \cdot d_j(n) \right\} \qquad (10)$$

and

$$\delta_k(n) = \min\left\{ \frac{d_{j^*}(n) - d_l(n)}{j^* - l}, \frac{1}{2} \right\}; \; \Lambda_k(n) = \arg\{E(t_k(\Lambda_k)) = \delta_k(n)\} \qquad (11)$$

for $k = l+1, \ldots, j^*$.
Checking Step: if $j^* = n$ then the procedure is completed. Otherwise, we put $l \leftarrow j^*$ and **Main Recursive Step** is repeated until $j^* = n$.

In Algorithm 1, the values of $\delta_1(n), \ldots, \delta_n(n)$ and $\Lambda_1(n), \ldots, \Lambda_n(n)$ are determined. Below, some of their features are analyzed under the following assumptions:

- $\delta_1(n) \leq \delta_2(n) \leq \ldots \leq \delta_n(n)$;
- $\sum_{j=1}^{i} \delta_j(n) \leq d_i(n)$. If $\delta_i(n) < \delta_{i+1}(n)$ then $\sum_{j=1}^{i} \delta_j(n) = d_i(n)$;
- $\Lambda_j(n) = \begin{cases} \sqrt{\frac{1}{6 \cdot \delta_j(n)}} & \text{if } 0 < \delta_j(n) \leq \frac{1}{6} \\ \frac{3}{2} - 3 \cdot \delta_j(n) & \text{if } \frac{1}{6} < \delta_j(n) \leq \frac{1}{2} \end{cases}, j = 1, \ldots, n;$
- $\Lambda_1(n) \geq \Lambda_2(n) \geq \ldots \geq \Lambda_n(n)$;
- If $\delta_j(n) = \delta_{j+1}(n)$ then $\Lambda_j(n) = \Lambda_{j+1}(n)$, $\lambda_j(n) = 0$, $E(s_j(\Lambda(n)) \leq d_j(n)$;
- if $\delta_j(n) < \delta_{j+1}(n)$ then $\Lambda_j(n) < \Lambda_{j+1}(n)$, $\lambda_j(n) > 0$, $E(s_j(\Lambda(n))) = d_j(n)$;
- $\sum_{i=1}^{n} \lambda_i(n) \cdot \left(\sum_{j=1}^{i} \delta_j(n)\right) = \sum_{i=1}^{n} \lambda_i(n) \cdot E(s_i(\Lambda(n))) = \sum_{i=1}^{n} \lambda_i(n) \cdot d_i(n)$;
- $E(p_j(\Lambda_j(n))) = \begin{cases} \sqrt{\frac{2}{3}} \cdot \delta_j(n) & \text{if } 0 < \delta_j(n) \leq \frac{1}{6} \\ \frac{1}{8} + \frac{3}{2} \cdot \delta_j(n) \cdot (1 - \delta_j(n)) & \text{if } \frac{1}{6} \leq \delta_j(n) \leq \frac{1}{2} \end{cases}$

Hence:

$$\begin{aligned} E(Z_n(\Lambda(n))) &= \sum_{j=1}^{n} E(p(\Lambda_j(n))) = \\ &= \sum_{j=1}^{n}\left(\tau_j(n)\left(\frac{1}{8}+\frac{3}{2}\delta_j(n)(1-\delta_j(n))\right)+\bar{\tau}_j(n)\sqrt{\frac{2\cdot\delta_j(n)}{3}}\right) \end{aligned}$$

where $\tau_j(n) = \begin{cases} 1 & \text{if } \frac{1}{6} < \delta_j(n) \le \frac{1}{2} \\ 0 & \text{otherwise} \end{cases}$ and $\bar{\tau}_j(n) = 1-\tau_j(n)$, $j = 1,\dots,n$.

The main result from Szkatuła [16] is stated in the theorem given below.

Theorem 1 *Let p_j, t_j, $j = 1,\dots,n$, be realizations of mutually independent random variables uniformly distributed over $(0,1]$, $d_1(n) \le d_2(n) \le \dots \le d_n(n)$, $d_j(n)$ be deterministic, and $\delta_j(n)$ be defined by the above Algorithm 1. If*

$$\frac{\ln(n)}{n\cdot\delta_1(n)} \approx 0$$

then:

$$z_{OPT}(n) \approx \sum_{j=1}^{n}\left(\tau_j(n)\left(\frac{1}{8}+\frac{3\cdot\delta_j(n)}{2}(1-\delta_j(n))\right)+\bar{\tau}_j(n)\sqrt{\frac{2\cdot\delta_j(n)}{3}}\right) \quad (12)$$

The main idea of the proof of Theorem 1 is based on showing that:

$$\hat{z}_n(\Lambda(n)) \approx E(z_n(\Lambda(n))) \approx \varphi_n(\Lambda(n)) \text{ and } E(z_n(\Lambda(n))) \approx z_n(\Lambda(n))$$

and using (6). For further details the Reader is kindly referred to Szkatuła [16].

4 Intervals of deadliness and the approximate algorithm

Construction of the $\delta_j(n)$, $j = 1,\dots,n$, provided by the Algorithm 1 and described in the Section 3, indicates that values of the $\delta_j(n)$ as well as their properties significantly depend on the mutual relations between $d_j(n)$, $j = 1,\dots,n$. The assumption that $d_j(n)$ are monotonic, cf. (1), is essential in the construction of the Algorithm 1 and, as the consequence, in Theorem 1. Formula (10) clearly indicates that values $\delta_1(n), \delta_2(n), \dots, \delta_n(n)$ depend on

$$d_1(n), d_2(n)-d_1(n), \dots, d_n(n)-d_{n-1}(n).$$

To be more precise, from the construction of $\delta_j(n)$, $\Lambda_j(n) = \sum_{i=j}^{n}\lambda_i(n)$, $j = 1,\dots,n$, it follows that:

$$\begin{aligned} &\text{if } \delta_j(n) = \delta_{j+1}(n) \text{ then } \Lambda_j(n) = \Lambda_{j+1}(n),\ \lambda_j(n) = 0,\quad E(s_j(\Lambda(n))) \le d_j(n); \\ &\text{if } \delta_j(n) < \delta_{j+1}(n) \text{ then } \Lambda_j(n) > \Lambda_{j+1}(n),\ \lambda_j(n) > 0,\quad E(s_j(\Lambda(n))) = d_j(n). \end{aligned}$$

From the duality theory it follows that if $\lambda_i(n) = 0$, then the corresponding constraint in (2) is "inactive". It means that its satisfaction is secured by other "active" constraints for which $\lambda_j(n) > 0$, $i, j \in \{1, \ldots, n\}$. Formula (10) makes it possible to distinguish three different classes of $\delta_j(n)$, $\Lambda_j(n)$, $\lambda_i(n)$, $i, j = 1, \ldots, n$.

Lemma 1 *If for all* $j = 2, \ldots, n$

$$d_{j-1}(n) \geq \frac{j-1}{j} \cdot d_j(n) \tag{13}$$

then

$$\delta_1(n) = \ldots = \delta_n(n);\ \Lambda_i(n) = \Lambda_{i+1}(n) = \lambda_n(n), \lambda_i(n) = 0, i = 1, \ldots, n-1. \tag{14}$$

Proof. If (13) holds, then in Algorithm 1, according to formula (10), Main Recursive Step is performed only once with $j^* = n$ and (14) will follow immediately from (11). ■

In the case considered in Lemma 1 only the last constraint is active and the SJD problem is reduced to the single-constraint knapsack problem (4).

Lemma 2 *If for all* $j = 2, \ldots, n$

$$d_{j-1}(n) < \frac{j-1}{j} \cdot d_j(n) \tag{15}$$

then

$$\delta_1(n) < \delta_2(n) < \ldots < \delta_n(n);\ \Lambda_1(n) > \Lambda_2(n) > \ldots > \Lambda_n(n),\ \lambda_i(n) > 0, \tag{16}$$

for $i = 1, \ldots, n$.

Proof. If (15) holds, then in Algorithm 1, according to formula (10), the Main Recursive Step is performed n times with $j^* = 1, 2, \ldots, n$, and (16) will follow immediately from (11). ■

Lemma 2 deals with the situation when all n constraints are active.

Lemma 3 *If there exists* $j\prime$, $1 < j\prime < n-1$ *such that*

$$d_{j-1}(n) \geq \frac{j-1}{j} \cdot d_j(n),\ j = 2, \ldots, j\prime;\ \textit{and}\ d_{j\prime+1}(n) < \frac{j\prime+1}{j\prime+2} \cdot d_{j\prime+2}(n) \tag{17}$$

then

$$\delta_1(n) = \ldots = \delta_{j\prime}(n) < \delta_{j\prime+1}(n);\ \Lambda_1(n) = \ldots = \Lambda_{j\prime}(n) > \Lambda_{j\prime+1}(n),\ \lambda_i(n) = 0, \tag{18}$$

where $i = 1, \ldots, j\prime$.

Proof. In this case, when (17) holds, then in Algorithm 1, according to formula (10), first execution of the Main Recursive Step yields $j^* = j\prime$. Then, starting with $l = j\prime + 1$, the Main Recursive Step will be performed at least once more and (18) will follow immediately from (11). ■

In Lemma 3 a mixed case is considered, where some, at least first $j\prime$, constraints are inactive, while some constraints, at least one $(j\prime + 1)$'th will be active. It also may happen that this situation will be repeated several times. For example, there may exist $j\prime\prime$ and $j\prime\prime\prime$, $j\prime < j\prime\prime < j\prime\prime\prime < n$, such that constraints $j\prime + 1, \ldots, j\prime\prime$ are active, constraints $j\prime\prime + 1, \ldots, j\prime\prime\prime$ are inactive and so on.

For any given set of deadlines $d_1(n), d_2(n), \ldots, d_n(n)$ Lemmas 1, 2 or 3 are covering all possible relations between deadlines and the resulting "activity" status of constraints. Therefore, these three lemmas allow for introducing recursive intervals of deadlines corresponding to the three cases considered. The theorem below presents the main result of this paper.

Theorem 2 *If for all $j = 2, \ldots, n-1$*

$$d_j(n) \in \left[\frac{j}{j+1} \cdot d_{j+1}(n), \frac{j}{j-1} \cdot d_{j-1}(n)\right] \tag{19}$$

holds, then Lemma 1 holds and (14) describes mutual relations between $\delta_j(n)$, $\Lambda_j(n)$, $\lambda_i(n)$, $i, j = 1, \ldots, n$. If for all $j = 2, \ldots, n-1$

$$d_j(n) \in \left(\frac{j}{j-1} \cdot d_{j-1}(n), \frac{j}{j+1} \cdot d_{j+1}(n)\right) \tag{20}$$

holds, then Lemma 2 holds and (16) describes mutual relations between $\delta_j(n)$, $\Lambda_j(n)$, $\lambda_i(n)$, $i, j = 1, \ldots, n$. If there exists $j\prime$, $2 < j\prime < n-1$ such that for all $j = 2, \ldots, j\prime - 1$

$$d_j(n) \in \left[\frac{j}{j+1} \cdot d_{j+1}(n), \frac{j}{j-1} \cdot d_{j-1}(n)\right] \text{ and } d_{j\prime+1}(n) < \frac{j\prime+1}{j\prime+2} \cdot d_{j\prime+2}(n) \tag{21}$$

holds, then Lemma 3 holds and (18) describes mutual relations between $\delta_j(n)$, $\Lambda_j(n)$, $\lambda_i(n)$, $i, j = 1, \ldots, n$.

Proof. In order to prove Theorem 2 it is sufficient to observe that Lemma 1 demonstrates (19), Lemma 2 demonstrates (20), and Lemma 3 demonstrates (21). ■

The results, summarized in Theorem 2 and Lemmas 1-3 go far beyond the results presented in Szkatuła [16], cf. Theorem 1. These results allow to analyze the impact of the mutual relations of the deadlines $d_j(n)$, $j = 1, \ldots, n$, on behaviour of the optimal solution $z_{OPT}(n)$ in the asymptotical case, i.e. when $n \to \infty$, for the considered random model of the SJD problem (2).

Formulas (19), (20) and (21) define in the recursive manner the deadline intervals. If deadlines $d_j(n)$, $j = 2, \ldots, n-1$, belong to the corresponding deadlines intervals, as presented in (19), (20) and (21), then it is guaranteed that $\delta_j(n)$, $\Lambda_j(n)$, $\lambda_i(n)$, $i, j = 1, \ldots, n$, will belong to one of the three different classes defined by Lemmas 1-3 and Theorem 2. This provides for certain flexibility in defining actual values of deadlines, since they may vary within the proposed deadline intervals. Because of the recursive definition of the deadline intervals, first and last deadlines may have special influence on the interval construction. The three examples, presented below, illustrate the above defined classes of $\delta_j(n)$, $\Lambda_j(n)$, $\lambda_i(n)$, $i, j = 1, \ldots, n$.

Example 1

Let

$$d_j(n) = \frac{j}{2} \text{ and then } \delta_j(n) = \frac{1}{2},\ j = 1, \ldots, n;\ d_n(n) = \frac{n}{2}.$$

In this case, assumptions (13) of Lemma 1 and of Theorem 1 are fulfilled and according to (12) we have:

$$z_{OPT}(n) \approx \frac{1}{2} \cdot n \tag{22}$$

which means that in this case only the last constraint is active, optimal solution has the maximum possible value and all jobs will be processed before their deadliness in the asymptotical case for the considered random model of the SJD problem.

Example 2

Let

$$d_j(n) = \frac{j^2}{2n} \text{ and then } \delta_j(n) = \frac{2j-1}{2n},\ j = 1, \ldots, n;\ d_n(n) = \frac{n}{2}.$$

In this case, assumptions (15) of Lemma 2 and of Theorem 1 are fulfilled. All constraints are active, all jobs will be processed before their deadlines and therefore (22) will hold.

Example 3

Let

$$\begin{aligned} d_j(n) &= \frac{j}{4} \text{ and then } \delta_j(n) = \frac{1}{4},\ j = 1, \ldots, n^*,\ n^* = \left\lceil \frac{n}{2} \right\rceil, \\ d_j(n) &= \frac{j^2}{2n} \text{ and then } \delta_j(n) = \frac{2j-1}{2n},\ j = n^*+1, \ldots, n;\ d_n(n) = \frac{n}{2}. \end{aligned}$$

In this case, assumptions (17) of Lemma 3, where $j\prime = n^*$, and of Theorem 1 are fulfilled. According to (11), $\delta_j(n) = \frac{1}{2}$, $j = n^* + 1, \ldots, n$, and from (12) we have:

$$z_{OPT}(n) \approx \frac{1}{2} \cdot n - \frac{3}{32} \cdot n^* - \frac{1}{2},$$

which means that in this case some constraints will be active, while some inactive, and the optimal solution value is smaller than the possible maximum one. Some jobs will not be processed before their deadlines (providing no profit) in the asymptotical case for the considered random model of the SJD problem.

An illustrative heuristic, greedy type algorithm, designed to solve the SJD problem (2) in the general deterministic case, is presented below. This algorithm is using the procedure equivalent to the one applied in the Algorithm 1.

Algorithm 2

Initialization Step: Let $l \leftarrow 0$, $d_0(n) \leftarrow 0$
Main Recursive Step: Let

$$j^* = \max_{l<m\leq n} \left\{ m \;\middle|\; \frac{d_m(n) - d_l(n)}{m-l} = \min_{l<j\leq n} \frac{d_j(n) - d_l(n)}{j-l} \right\}$$

and

$$\Lambda_{j^*} = \min_{\Lambda \in \Phi_{j^*}} \left\{ \Lambda \;\middle|\; \sum_{j=l+1}^{j^*} t_j(\Lambda) \leq d_{j^*}(n) \right\}; \Phi_{j^*} = \left\{ \frac{p_{l+1}}{t_{l+1}}, \ldots, \frac{p_{j^*}}{t_{j^*}} \right\} \qquad (23)$$

$$x_j \leftarrow x_j(\Lambda_{j^*}),\; j = l+1, \ldots, j^*, \text{ refer to (5).}$$

Checking Step: if $j^* = n$ then the Algorithm 2 is completed. Otherwise, we put $l \leftarrow j^*$ and the **Main Recursive Step** is repeated until $j^* = n$.

Algorithm 2 has extremely low computational complexity of order of $O(n)$. It does not require sorting of elements as greedy type algorithms usually do. From the computational point of view, the most expensive are max and min operations, which have computational cost of $O(k)$, where k is the number of elements of the corresponding set. In the case when max and min operations are repeated k times over sets of order of k elements, the computational complexity will be of the order of $O(k^2)$. According to Lemmas 1 - 3, this situation cannot occur, since either the Main Recursive Step will be executed only once (Lemma 1) or it will be executed several times (n times in the case of Lemma 2 and the maximum of $n-1$ times in the case of Lemma 3) but the number of elements of the corresponding sets will be small, i.e. of order of $\frac{n}{m}$, where n is the number of jobs (i.e. the size of the problem) and m is the number of necessary repetitions of the Main Recursive Step. Therefore, the overall computational complexity of Algorithm 2 will be of the order of $O(n)$.

In the sense of the worst case analysis this algorithm always provides feasible solutions of the SJD problem (2), which is guaranteed by (23), but it does not have any guarantees of the accuracy of the solutions provided. Therefore, in the sense of the worst case analysis Algorithm 2 is an heuristic algorithm. However, for the random model of the problem (2), considered in this paper, Algorithm 2 is asymptotically sub-optimal. In the random case Algorithm 2 is behaving identically to the computational procedure described in the Algorithm 1. Asymptotical suboptimality of Algorithm 2 follows immediately from Theorems 1 and 2. The proposed algorithm is based on the main results of this paper, presented in the Lemmas 1-3 and Theorem 2, and it has mainly an illustrative purpose.

A commonly used practice in testing of algorithms proposed for job sequencing or scheduling problems is to generate a number of random problems, whose coefficients are usually random integers from the intervals [0,100] or [0,10], cf. El-Bouri et al. [6], Sevaux and Dauzere-Péres [15], Baptiste and Le Pape [2], Baptiste et al. [1], Detienne [5], Levner and Elalouf [12], and Catanzaro et al. [4], just to mention a few representative examples. Behaviour of the SJD problem solutions, in the light of the results provided by the probabilistic analysis, may serve as the proper substitution for the computational tests, e.g. in the case of Algorithm 2, thus saving time and computational effort.

5 Concluding remarks

In the present paper one of the classical scheduling problems - Sequencing Jobs with Deadlines problem (SJD) was considered. Probabilistic analysis of the impact of mutual relations between deadlines and their functional properties was performed for the random model of the SJD problem. As the result of this analysis, three specific categories of mutual relations between deadlines were identified. Then, on the basis of the results provided in Lemmas 1, 2, and 3, Theorem 2 was formulated, in which recursive deadline intervals for the considered random model of the SJD problem were defined. In this framework, the roles of the first and last constraints, $d_1(n)$ and $d_n(n)$, respectively, are crucial. Deadline intervals may provide substantial flexibility in formulating SJD problems, because mutual relations between deadlines may be analyzed in a more convenient manner and their influence on the SJD problem solutions (the list of the jobs to be performed before their deadlines and providing maximized profit) is demonstrated in the convenient and convincing manner.

Another interesting result of the study, reported in the paper, is a simple heuristic algorithm of very low computational complexity, which in the average case, i.e. for the considered random model of the SJD problem, is asymptotically sub-optimal.

These results are enriching the knowledge base for the Scheduling Problems, especially for the Sequencing Jobs with Deadlines problem and may also positively influence the solvability of the SJD problem instances in practical applications.

Acknowledgements

The research reported was partially supported by the National Science Center Poland (NCN), decision no.: DEC-2011/03/B/ST6/00364

The author wants to thank the anonymous referees for valuable comments and suggestions, which have allowed to significantly improve the present version of the paper.

References

[1] P. Baptiste, F. D. Croce, A. Grosso, and V. Tkindt. Sequencing a single machine with due dates and deadlines: an ilp-based approach to solve very large instances. *Journal of Scheduling*, 13:39–47, 2010.

[2] P. Baptiste and C. Le-Pape. Scheduling a single machine to minimize a regular objective function under setup constraints. *Discrete Optimization*, 2:83–99, 2005.

[3] J. Błażewicz, K. Ecker, E. Pesch, G. Schmidt, and J. Weglarz. *Scheduling Computer and Manufacturing Processes.* Springer Verlag, 1996.

[4] D. Catanzaro, L. Gouveiac, and M. Labb. Improved integer linear programming formulations for the job sequencing and tool switching problem. *European Journal of Operational Research*, 244:766–777, 2015.

[5] B. Detienne. A mixed integer linear programming approach to minimize the number of late jobs with and without machine availability constraints. *European Journal of Operational Research*, 235:540552, 2014.

[6] A. El-Bouri, S. Balakrishnan, and N. Popplewell. Sequencing jobs on a single machine: A neural network approach. *European Journal of Operational Research*, 126:474–490, 2000.

[7] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, 1979.

[8] A. Janiak, W. A. Janiak, T. Krysiak, and T. Kwiatkowski. A survey on scheduling problems with due windows. *European Journal of Operational Research*, 242:347–357, 2015.

[9] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems.* Springer Verlag, 2004.

[10] G. Kocur. *Courses on Computer Algorithms in Systems Engineering, Greedy algorithms (lecture 10): knapsack, job sequence.* Massachusetts Institute of Technology: MIT OpenCourseWare, 2010.

[11] E. Lawler and J. Moore. A functional equation and its application to resource alocation and sequencing problems. *Management Science*, 16:77–84, 1969.

[12] E. Levner and A. Elalouf. An improved approximation algorithm for the ancient scheduling problem with deadlines. In *Control, Decision and Information Technologies (CoDIT), 2014 International Conference on*, 2014.

[13] A. Puntambekar. *Analysis of Algorithm and Design.* Technical Publications, 2009.

[14] S. Sahni. Algorithms for scheduling independent jobs. *Journal of ACM*, 23:116–127, 1976.

[15] M. Sevaux and S. Dauzère-Pérès. Genetic algorithms to minimize the weighted number of late jobs on a single machine. *European Journal of Operational Research*, 151:296306, 2003.

[16] K. Szkatuła. Random sequencing jobs with deadlines problem: growth of the optimal solutions values. *European Journal of Operational Research*, 109:160–169, 1998.

[17] Y. Xia, B. Chen, and J. Yue. Job sequencing and due date assignment in a single machine shop with uncertain processing times. *Discrete Optimization*, 184:6375, 2008.