Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Invited Review

A survey on scheduling problems with due windows

Adam Janiak[a,*], Władysław A. Janiak[b], Tomasz Krysiak[a], Tomasz Kwiatkowski[a]

[a] Systems Research Institute, Polish Academy of Sciences, Newelska 6, 01-447 Warsaw, Poland
[b] The International University of Logistics and Transport in Wrocław, Sołtysowicka St. 19B, 51-168 Wrocław, Poland

ARTICLE INFO

Article history:
Received 29 November 2012
Accepted 21 September 2014
Available online 29 October 2014

Keywords:
Scheduling
Due windows
Due window assignment
Earliness/tardiness
Number of early and tardy jobs

ABSTRACT

In this paper, a survey of scheduling problems with due windows is presented. The due window of a job is a generalization of the classical due date and it is a time interval in which this job should be finished. If the job is completed before or after the due window, it incurs an earliness or a tardiness penalty. A review of an extensive literature concerning problems with various models of given due windows, due window assignment and job-independent and job-dependent earliness/tardiness penalties is presented. The article focuses mainly on the computational complexity of the problems, mentioning also their solution algorithms. Moreover, the practical applications of the reviewed problems are mentioned, giving the appropriate references to the scientific literature. One particularly interesting IT example is described in detail.

1. Introduction

Scheduling problems with due windows have their origins in Just-in-Time (JIT) philosophy. The main objective of JIT is to ensure that the required material is available exactly when it is needed, while maintaining the inventory at the lowest possible level, eliminating such waste components like overproduction, waiting time, unnecessary transportation, unnecessary processing, defective products, etc. (Kramer & Lee, 1993). Usually, JIT scheduling models assume an existence of job due dates and penalize both early and tardy jobs. If a job is completed before its due date, it implies earliness penalty, such as holding cost. On the other hand, a job finished after its due date implies tardiness cost, such as late charges, express delivery, or loss of sales. The recent results on JIT scheduling models with due dates can be found, e.g., in Shabtay (2012), Sourd (2005), Esteve, Aubijoux, Chartier, and T'kindt (2006), Ventura and Radhakrishnan (2003), Tuong and Soukhal (2010), Prot, Bellenguez-Morineau, and Lahlou (2013), Rasti-Barzoki and Hejazi (2013). For the earlier surveys on the problems with due dates see also Baker and Scudder (1990), Cheng and Gupta (1989), Gordon, Proth, and Chu (2002) and Lauff and Werner (2004). However, in manufacturing industry it is often expected that the jobs are finished in a certain time interval (due window) rather than at single point in time (due date) (Kramer & Lee, 1993). A due window is a generalization of the due date concept and defines a time interval in which a job should be finished. If the job is completed within the due window, then it is considered to be on time and no penalty is incurred. Otherwise it incurs an earliness or a tardiness penalty, depending on whether the job was completed before or after the due window, respectively.

The scheduling problems with due dates as well as the ones with due windows are perceived as bicriteria or even multicriteria optimization problems. Namely, most of them are the problems of minimizing a linear composite objective function in earliness penalties, tardiness penalties and in some cases position of due dates or position and size of due windows (see e.g. Hoogeveen 2005 for details). Some of these problems consist in minimizing one objective (e.g., the sum of earliness penalties) subject to some constraints on other objectives (e.g., the sum of tardiness penalties), see e.g., Wan and Yen (2009), Cheng, Tadikamalla, Shang, and Zhang (2014).

There is a variety of applications of due window scheduling models in JIT manufacturing (Wu & Lai, 2007), semi-conductor manufacturing (Lee, 1991), chemical processing, PERT/CPM scheduling (Koulamas, 1996) as well as in IT. Consider, for instance, an IT company that provides access to multimedia data via Internet. Modern Internet services allow users to access multimedia data without downloading the whole file beforehand. An example of such a service is Video on Demand (VoD). Time constraints (required to ensure fluent and continuous data presentation) are very important in transmission of multimedia data. Constant miniaturization of mobile devices and the increase in popularity of high resolution video are the reasons, why not only the tardiness, but also earliness of the data transmission should be taken into account. This is connected with storing large amount of data in the mobile device's input buffer (i.e., specially allocated part of the device memory). Namely, too early data transmission can cause overflow of this buffer. In such case, part of the data is lost

* Corresponding author. Tel.: +48 71 320 29 06; fax: +48 71 321 26 77.
E-mail addresses: adam.antoni.janiak@gmail.com (A. Janiak), wladyslaw.janiak@gmail.com (W. A. Janiak), tomasz.a.krysiak@gmail.com (T. Krysiak), tomasz.1.kwiatkowski@gmail.com (T. Kwiatkowski).

http://dx.doi.org/10.1016/j.ejor.2014.09.043
0377-2217/© 2014 Elsevier B.V. All rights reserved.

and retransmission is necessary to keep the continuity of the data presentation. In an ideal case the data is delivered exactly at the moment when it needs to be displayed.

User's device (computer, tablet or smartphone) initiates communication with a server, demanding the start of various data transmission. The multimedia data is stored on the server hard disk drive and sent to each user's device in fragments, usually in RTP (Real-time Transport Protocol (Schulzrinne, Casner, Frederick, & Jacobson, 2003)) packets (Perkins, 2003). The data fragments must be properly prepared (e.g. compressed, encrypted) on the server before they can be sent. Afterward, they are transmitted via server's network interface. The user's device stores the data packets in its buffer and creates subsequent video/audio frames by ordering, decrypting and decompressing the data packets. The order and time at which the data fragments are sent to the users' devices are established according to the schedule created by the server's software.

Depending on the moment in which a given data fragment is sent by the server, three different situations are possible. First, and the most desirable is the one in which the user's device buffer has enough free space to store the received fragment and there is still some time left before the video/audio frame is to be presented. This causes no interruption in the data presentation. In the second situation, the data fragment is received after the frame was to be presented, which obviously disrupts the presentation. In the third situation, the data is received too early which causes the buffer overflow, loss of the data packet and need for retransmission. Notice also that the device may increase the size of the buffer, causing additional software load that may lead to disruptions in the audio/video data being presented at the time. Either way, the customer may feel dissatisfied with an imperfect service (i.e., by disruptions). As a result, the provider can lose such a customer and have its profit lower than expected.

The problem of data fragments scheduling described above is to assign each data fragment to one of the server's processors and to determine the moments at which each fragment processing is to be finished. The objective is to ensure that all the data fragments are prepared for sending to users and that the total cost associated with the late deliveries, buffers reallocations and retransmission of these fragments is minimized.

The above situation is reflected by the following scheduling problem with due windows. All the necessary actions performed by the server and needed to prepare the data fragment for sending are referred as a *job*. Each job is to be processed on one of the *parallel processors*. The *job processing time* is the time needed to perform the data processing actions, i.e., reading the data from the hard disk drive, preparation (i.e., compression, encryption, etc.) and sending them to the network interface. This time is proportional to the size of the data fragment.

The beginning of the due window is the earliest moment at which the data packet can be sent to the mobile device and does not cause the buffer overflow. The end of the due window is the latest moment at which the data fragment can be sent and does not disrupt the presentation. These moments can be calculated based on the capacity and load of the user's device buffer at the moment t (i.e., the amount of data stored in the buffer expressed in bytes), the speed of data reading from this buffer (in bytes per second), the amount of data (in bytes) sent from the server to the buffer after the moment t, the total amount of memory of the user's device (in bytes), by which the buffer should be resized as a result of requests sent by the server, RTT (*Round Trip Time*, i.e., the amount of time, in seconds, needed to exchange the above mentioned control information). Note that these control information may be exchanged using the following RTCP (Real-time Transport Control Protocol) packets: Sender Report RTCP packets, Receiver Report RTCP packets and Application-defined RTCP packets (see Esteve et al. 2006 for details).

The earliness penalty relates to the cost of the buffer resizing in the user's device. The tardiness penalty is related to the cost of disruption of the presentation in the user's device. Both above mentioned penalties are proportional to the charge for using the service by the customers (since in both cases the service becomes unreliable which can cause loss of dissatisfied customers). The exact value of these penalties should be determined by statistical analysis of the real data transmissions. The whole above example was originally presented in Januszkiewicz (2012).

Notice that in the scheduling literature there also exist several other concepts of executing jobs in time intervals (windows). However, these concepts are more restrictive than the idea of due windows, i.e., job execution is forbidden outside these fixed (or optimized) intervals. The most restrictive seem to be the *interval scheduling problems*, in which there is given a set of available parallel processors and all jobs have fixed starting and completion times. The aim is to determine whether there exists a feasible non-preemptive schedule for all jobs (Kroon, Salomon, & Van Wassenhove, 1995), or to minimize the total number of processors needed to process all jobs (Lee, Turner, Daskin, Homem-de-Mello, & Smilowitz, 2012), or (in the case in which not all jobs need to be assigned) to maximize the total profit of accepted jobs (or minimize the total cost of rejected jobs) (Kovalyov, Ng, & Cheng, 2007; Spieksma, Papadimitriou, Kolen, & Lenstra, 2007).

Less restrictive are scheduling problems with *minimum and maximum time lags*, where each job has a given interval (described by minimum and maximum time lags), in which it has to be started (this is not a time point like in the previous problems, but an interval). These time lags are associated with completion time of preceding job Sheen and Liao (2007).

If the intervals are not associated with the preceding jobs, then such problems are known as *scheduling problems with time windows* (Sedeño-Noda, Alcaide, & González-Martín, 2006; Sedeño-Noda, Alcaide López de Pablo, & González-Martín, 2009). For the problems with time windows – similarly as for the previously mentioned problems – the jobs cannot be executed outside the time windows. Such problems are sometimes referred to as the problems with *hard time windows*, in contrast to the problems with *soft time windows*. In the latter problems it is allowed to execute jobs outside the time windows to some extent (which is obviously penalized in appropriate way) (Fagerholt, 2001). Thus, the concept of the soft time windows are very similar to the concept of the due windows. However, the soft time windows are used rather in the problems from transportation and logistics area (ship scheduling, vehicle routing, etc.), where the job scheduling is only a part of more complex optimization processes. For this reason, the problems with soft time windows are out of the scope of our survey paper.

The purpose of this paper is to present the main features of approaches developed to solve scheduling problems with due windows. We concentrate on the computational complexity results for the considered problems and the proposed solution algorithms. The remainder of the paper is organized as follows. In the next section we present the three-field notation for the scheduling problems. Section 3 concerns scheduling problems with given due windows. In Section 4 we describe results for problems with a common due window assignment. Section 5 concentrates on scheduling problems with other models of due window assignment. We conclude the paper in Section 6.

2. Problem statement, definitions and notation

Now we present the notation which is used in the paper. We examine problems of scheduling n jobs $(J_1, J_2, \ldots, J_n)$ on m (mainly parallel) processors $(M_1, M_2, \ldots, M_m)$ to minimize a given objective function. It is assumed that a processor can process at most one job at a time and all jobs are ready for processing at time $t = 0$. Each job $j, j = 1, \ldots, n$, is characterized by a set of m processing times $p_{ij}, i = 1, \ldots, m$ (i.e., there are different processing times on different