Computers & Operations Research 41 (2014) 115–124

Contents lists available at ScienceDirect

Computers & Operations Research

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

Batch scheduling of identical jobs with controllable processing times

Baruch Mor [a], Gur Mosheiov [b,*]

[a] The Department of Economics and Business Administration, Ariel University, Israel
[b] School of Business Administration, The Hebrew University, Jerusalem, Israel

ARTICLE INFO

Available online 20 August 2013

Keywords:
Batch scheduling
Controllable processing times
Flowtime
Single machine
Parallel identical machines

ABSTRACT

In scheduling models with *controllable processing times*, the job processing times can be controlled (i.e. compressed) by allocating additional resources. In *batch scheduling* a large number of jobs may be grouped and processed as separate batches, where a batch processing time is identical to the total processing times of the jobs contained in the batch, and a setup time is incurred when starting a new batch.

A model combining these two very popular and practical phenomena is studied. We focus on identical jobs and linear compression cost function. Two versions of the problem are considered: in the first we minimize the sum of the total flowtime and the compression cost, and in the second we minimize the total flowtime subject to an upper bound on the maximum compression. We study both problems on a single machine and on parallel identical machines. In all cases we introduce closed form solutions for the relaxed version (allowing non-integer batch sizes). Then, a simple rounding procedure is introduced, tested numerically, and shown to generate extremely close-to-optimal integer solutions. For a given number of machines, the total computational effort required by our proposed solution procedure is $O(\sqrt{n})$, where n is the number of jobs.

1. Introduction

Vickson [1,2] introduced a new class of scheduling models with *controllable processing times*. In this model the job processing times are not given constants as in classical scheduling, but can be controlled (i.e. compressed) by allocating additional resources. Various versions of this very practical scheduling setting have been studied by many researchers, as reflected in the recent survey of Shabtai and Steiner [3]. Some of the early papers are: Van Wassenhove and Baker [4], Nowicki and Zdrzalka [5], Janiak and Kovalyov [6], Wan et al. [7], Hoogeveen and Woeginger [8], Janiak et al. [9], Shakhlevich and Strusevich [10], Wang [11] Akturk et al. [12] and Wang and Xia [13]. More recently, Tseng et al. [14] studied a single machine setting with controllable processing times and an objective of minimum total tardiness; Turkcan et al. [15] studied a setting of parallel machines and objective functions of minimum manufacturing cost and total weighted earliness and tardiness; Shakhlevich et al. [16] focused on the trade-off between the maximum cost (which is a function of the completion times) and the total compression cost; Shabtay et al. [17] addressed due date assignment problems in a group technology environment; Gurel et al. [18] considered failures of the machine and repair time, and focused on an anticipative scheduling approach; Wan et al. [19] studied the problem of scheduling jobs of two-agents on a single machine with controllable processing times; Choi et al. [20] focused on minimizing weighted completion time subject to an upper bound on the maximum compression cost; Leyvand et al. [21] considered just-in-time scheduling on parallel machines; Yin and Wang [22] studied a model combining controllable processing times and learning; Wang and Wang [23] addressed the single machine problem of minimizing the total resource consumption subject to an upper bound on the total weighted flowtime; Wei et al. [24] focused on a model in which the job processing times are a function of both resource consumption and the job starting times; Uruk et al. [25] studied a two-machine flowshop problem with flexible operations and controllable processing times; and Wang and Wang [26] focused on convex resource dependent processing times and job deterioration.

In *batch scheduling* a large number of jobs may be grouped and processed as separate batches. Such batching is generally based on the existence of some similarity between jobs belonging to the same class. A batch processing time is identical to the total processing times of the jobs contained in the batch. A *setup time* is incurred when starting a new batch. In their classical paper, Santos and Magazine [27] solved a single machine batch scheduling problem to minimize total flowtime. They assumed a constant (i.e. batch independent) setup time, and *batch availability*, i.e., jobs

* Corresponding author.
E-mail address: msomer@mscc.huji.ac.il (G. Mosheiov).

0305-0548/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2013.08.007

are completed only when their entire batch is completed. Dobson et al. [28], Naddef and Santos [29] and Coffman et al. [30] extended the basic results obtained by Santos and Magazine for the "relaxed version" of the problem (when integer batch sizes are not required). Later, Shallcross [31] and Mosheiov et al. [32] solved the integer version. We refer the reader also to the extensive survey of Allahverdi et al. [33] on batch scheduling under different machine settings and objective functions.

In this paper we combine (to our knowledge for the first time) the two phenomena of batch scheduling and controllable processing times. Specifically, with respect to batching, we consider the setting of Santos and Magazine [27], i.e. a single machine, batch independent processing times, batch availability and the objective of minimum flowtime. With respect to the option of controlling job processing times, we assume *linear* compression cost. Two problems are solved: in the first we minimize the total flowtime plus the compression cost, and in the second we minimize flowtime, subject to an upper bound on the maximum batch compression. We first focus on a single machine setting, and show that in both cases, the solution for the relaxed version consists of a decreasing arithmetic sequence of batch sizes, for which closed form solutions are obtained. The total running time of the procedure is $O(\sqrt{n})$, where n is the number of jobs. An integer solution is obtained by a simple rounding procedure, requiring additional $O(\sqrt{n})$ time. Our numerical tests indicate that the integer solutions are extremely close to those of the relaxed versions (which are lower bounds on the optimal solutions). We then consider the setting of parallel identical machines. [We refer the reader to Mor and Mosheiov [34], who studied batch scheduling on parallel identical machines without the option of compression.] In this case, we show that the solution of both problems consists of identical decreasing arithmetic sequences of the batch sizes on all the machines. The total computational effort of this more complicated solution procedure becomes $O(m\sqrt{n})$, where m is the number of machines. A rounding procedure and its evaluation are provided for this setting as well.

In Section 2 we present the notation and the formulation. In Sections 3 and 4, we solve the relaxed versions of the two different problems on a single machine. In Section 5 we propose the rounding procedure to obtain an integer solution for both problems. Sections 6 and 7 are devoted to numerical examples and numerical tests, respectively. Section 8 contains the extension to parallel identical machines.

2. Notation and formulation

We consider n identical jobs which need to be processed on a single machine. Jobs may be processed in batches, sharing the same setup operation: when starting a new batch, a setup time, denoted by s ($s>0$) is performed. For a given job allocation to batches, we denote by K the total number of batches. By allocating certain resources, batch sizes can be compressed from their maximum (original) size denoted by $\overline{n_j}$, down to their final batch size denoted by $n_j>0$, $j=1,\dots,K$. (For simplicity we assume in the following that all the jobs have unit processing times prior to compression. It follows that $\sum_{j=1}^{K}\overline{n_j}=n$.) Let N_c denote the total amount of compression of all K batches. Clearly, $n=\sum_{j=1}^{K}n_j+N_c$. The unit compression cost is denoted by $c>0$, thus the total compression cost is given by cN_c.

The first objective function considered here is the sum of the total flowtime and the batch compression cost. As mentioned above, we assume *batch availability*, i.e., the completion time of a job is defined as the completion time of the batch to which it is assigned. Thus, for a given allocation of jobs to batches, let C_j denote the completion time of batch j, $j=1,\dots,K$. The contribution of batch j to the total flowtime is n_jC_j, implying that the total flowtime is given by $\sum C=\sum_{j=1}^{K}n_jC_j$. Thus, the objective function of the problem is $\sum C+cN_c$, and using the conventional notation, the first problem studied here (P1) is

$$\text{P1}: \ 1/batch,\ \ ctrl/\textstyle\sum C+cN_c.$$

While P1 is a legitimate objective function (and clearly any extension to a linear combination of the above two cost components), theoretically, it may lead to an extreme non-realistic solution, where the batch sizes are compressed to zero. One way to handle this feasibility issue is by limiting the maximum allowable compression. Thus, for the second problem (P2), we define an upper bound U on the maximum compression. In order to guarantee a realistic solution, we restrict U to be strictly smaller than n. The objective function here is minimum flowtime, subject to this upper bound on the maximum compression

$$\text{P2}: \ 1/batch,\ \ ctrl,\ N_c\le U/\textstyle\sum C.$$

We extend both models to a setting of m parallel identical machines. For a given job allocation to the machines, let K_i denote the number of batches processed on machine i, $i=1,\dots,m$. Let $n_{i,j}$ denote the size of batch j processed on machine $i, i=1,\dots,m,\ \ j=1,\dots,K_i$. Thus, the second part of the paper consists of the following two problems (P3 and P4, respectively):

$$\text{P3}: \ Pm/batch,\ \ ctrl/\textstyle\sum C+cN_c.$$

$$\text{P4}: \ Pm/batch,\ \ ctrl,\ N_c\le U/\textstyle\sum C.$$

3. Problem P1: $1/\boldsymbol{batch},\ \boldsymbol{ctrl}/\sum \boldsymbol{C}+\boldsymbol{cN_c}$

In this section we provide a formal definition of the problem, present the basic properties of an optimal solution, and introduce closed form expressions for the optimal batch sizes. The combined objective function (flowtime plus compression cost) is given by

$$f_1=(s+n_1)n_1+(2s+n_1+n_2)n_2+\cdots+\left((K-1)s+\sum_{j=1}^{K-1}n_j\right)n_{K-1}+\left(Ks+\sum_{j=1}^{K}n_j\right)n_K+cN_c.$$

$$f_1=\sum_{j=1}^{K}\left(\sum_{i=1}^{j}n_i\right)n_j+s\sum_{j=1}^{K}jn_j+cN_c.$$

It is easy to verify the following equality:

$$\sum_{j=1}^{K}\left(\sum_{i=1}^{j}n_i\right)n_j=\frac{1}{2}\sum_{j=1}^{K}n_j^2+\frac{1}{2}\left(\sum_{j=1}^{K}n_j\right)^2.$$

Thus, we get the following objective function:

$$f_1=\frac{1}{2}\sum_{j=1}^{K}n_j^2+\frac{1}{2}\left(\sum_{j=1}^{K}n_j\right)^2+s\sum_{j=1}^{K}jn_j+cN_c. \tag{1}$$

The formal presentation of the problem is

$$\begin{aligned}&\text{Min}\ \ f_1\\ &\text{s.t.}\ \ \sum_{j=1}^{K}n_j+N_c=n\\ &\quad N_c\ge 0,\\ &\quad n_j\ge 0,\quad j=1,\dots,K.\end{aligned}$$

Clearly f_1 is a quadratic convex function of the batch sizes, implying that the global minimum can be found by applying the Karush–Kuhn–Tucker (KKT) conditions. The Lagrangian, with λ being the single Lagrange multiplier, is

$$L=f_1-\lambda\left(\sum_{j=1}^{K}n_j+N_C-n\right).$$