Computers & Industrial Engineering 92 (2016) 16–30

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Two-agent single-machine scheduling to minimize the batch delivery cost

Yunqiang Yin [a,*], Yan Wang [a], T.C.E. Cheng [b], Du-Juan Wang [c], Chin-Chia Wu [d,*]

[a] Faculty of Science, Kunming University of Science and Technology, Kunming 650093, China
[b] Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
[c] School of Management Science and Engineering, Dalian University of Technology, Dalian 116023, China
[d] Department of Statistics, Feng Chia University, Taichung, Taiwan

ARTICLE INFO

Article history:
Received 15 May 2015
Received in revised form 28 October 2015
Accepted 4 December 2015
Available online 14 December 2015

Keywords:
Scheduling
Batch delivery
Two agents
Fully polynomial-time approximation scheme

ABSTRACT

We consider integrated production and batch delivery scheduling in a make-to-order production system involving two competing agents, each of which having its own job set competes to process its jobs on a shared single machine. To save the delivery cost, the jobs of the same agent can be processed and delivered together batches. The completion time of each job in the same batch coincides with the batch completion time. A batch setup time is incurred before the processing of the first job in each batch. Each of the agents wants to minimize an objective function depending on the completion times of its own jobs. The goal is to determine a schedule for all the jobs of the two agents that minimizes the objective function of one agent, while keeping the objective function value of the other agent below or at a given value. For each of the problems under consideration, we either provide a polynomial-time algorithm to solve it or show that it is $\mathcal{NP}$-hard. In addition, for each of the $\mathcal{NP}$-hard problems, we present a mixed integer linear programming (MILP) formulation and provide a pseudo-polynomial dynamic programming algorithm, establishing that it is $\mathcal{NP}$-hard in the ordinary sense only, and show that it admits an efficient fully polynomial-time approximation scheme, if viable. Finally, we compare the performance of the pseudo-polynomial dynamic programming algorithms against the corresponding MILP formulations with randomly generated instances.

1. Introduction

We consider the following scheduling problem: There are two competing agents (A and B) each having its own set of independent and non-preemptive jobs to process on a shared single machine. Agent A has to process the job set $J^A = \{J_1^A, J_2^A, \ldots, J_{n_A}^A\}$, whereas agent B has to process the job set $J^B = \{J_1^B, J_2^B, \ldots, J_{n_B}^B\}$. All the jobs are available for processing from time zero onwards. The jobs belonging to J^A (resp., J^B) are called the *A-jobs* (resp., *B-jobs*). Associated with each job J_j^A (resp., J_j^B), there are a processing time p_j^A (resp., p_j^B), a weight w_j^A (resp., w_j^B), and a due date d_j^A (resp., d_j^B). All the data are assumed to be nonnegative integers. Let $n = n_A + n_B$ represent the total number of jobs and $P = P^A + P^B$ denote the total processing time of all the jobs, where $P^A = \sum_{j=1}^{n_A} p_j^A$ and $P^B = \sum_{j=1}^{n_B} p_j^B$.

The jobs are processed and delivered in batches. A batch can contain jobs only of the same agent, and a batch is referred to as an *A-batch* (resp., *B-batch*) if it contains only the *A*-jobs (resp., *B*-jobs). Each *A-batch* (resp., *B-batch*) has a sequence-independent batch setup time s_A (resp., s_B) and batch delivery cost φ_A (resp., φ_B) independent of the number of jobs in the batch. We adopt the *serial*-batching or *s*-batch assumption, whereby all the jobs in a batch are considered to have been completed together at the completion time of the last job in the batch, i.e., the processing time of a batch is equal to the total processing time of the jobs in the batch, and the completion time of each job in a batch is defined as the completion time of the batch. For the sake of simplicity, we assume that batch delivery is instantaneous.

For any schedule, denote the completion time of job J_j^A (resp., J_j^B) by C_j^A (resp., C_j^B). Each agent wants to minimize an objective function depending on the completion times of its jobs only. The two objective functions may be the same or different. We focus on the constrained optimization problem where the objective function value of agent A has to be minimized, whereas the objective function value of agent B must be kept less than or equal to a given

* Corresponding authors.
E-mail addresses: yinyunqiang@126.com (Y. Yin), cchwu@fcu.edu.tw (C.-C. Wu).

http://dx.doi.org/10.1016/j.cie.2015.12.003
0360-8352/

value U. Following Agnetis, Mirchandani, Pacciarelli, and Pacifici (2004), we denote the scheduling problem under study as $\alpha|\beta|\gamma^A:\gamma^B$, where γ^A and γ^B denote the objective functions of agents A and B, respectively. Specifically, we consider the following problems: $1|s-batch|\sum C_j^A+m_A\varphi_A:L_{\max}^B+m_B\varphi_B$, $1|s-batch|\sum C_j^A+m_A\varphi_A:\sum C_j^B+m_B\varphi_B$, $1|s-batch|\sum C_j^A+m_A\varphi_A:\sum w_j^BU_j^B+m_B\varphi_B$, and $1|s-batch|\sum w_j^AU_j^A+m_A\varphi_A:\sum w_j^BU_j^B+m_B\varphi_B$, where $L_{\max}^B=\max_{j=1,2,\ldots,n_B}\{C_j^B-d_j^B\}$, $U_j^A=1$ (resp., $U_j^B=1$) if $C_j^A>d_j^A$ (resp., $C_j^B>d_j^B$) and $U_j^A=0$ (resp., $U_j^B=0$) otherwise, and m_A (resp., m_B) denotes the number of batches for agent A (resp., B).

The following practical example concerning a two-level supply chain that involves a manufacturer and two customers (i.e., agents), namely agents A and B, motivates the scheduling problem under study. In this example, the manufacturer, referred to as a single "machine", receives from agents A and B two sets of customer orders, referred to as "jobs". A sufficient number of vehicles are available to deliver the finished orders to the customers, where each shipment can take any number of jobs and the cost per delivery is fixed. To save the delivery cost, the finished orders of each customer are delivered in batches. The processing requirement dictates that the orders belonging to the same batch (a delivery shipment) are processed contiguously and the dispatch date of a batch is equal to the completion time of the last order in the batch. A setup time is needed to perform some cleaning operations, or remove a previous container and install a new one when the manufacturer switches processing from one batch to another batch. The scheduling criteria involve the two customers' service levels, which are evaluated by different performance measures depending on the two customers' preferences. This situation can be modeled as our scheduling problem.

1.1. Literature review

The general problem under consideration in this paper falls into the following two categories of scheduling problem, namely batch delivery scheduling and multi-agent scheduling.

Batch scheduling, which combines job sequencing and partitioning, has attracted much attention of scheduling researchers in recent years. Potts and Kovalyov (2000) survey the batching issues in scheduling. However, classical batch scheduling research treats the delivery cost as either negligible or irrelevant. In other words, it focuses on machine scheduling, while ignoring job delivery scheduling. Production and distribution operations are two key operational functions in a supply chain. To achieve optimal operational performance in a supply chain, it is crucial to integrate these two functions, and plan and schedule them jointly in a coordinated manner (Chen, 2010). First to consider batch delivery scheduling, Cheng, Ng, and Yuan (2008) study a single-machine batch delivery scheduling problem to minimize the sum of the total weighted earliness (the difference between a job's delivery date and completion time) and delivery cost. They show that the problem is $\mathcal{NP}$-hard in the ordinary sense, and the case of the problem where the weights are equal is polynomially solvable. Hall and Potts (2003) study a variety of scheduling, batching, and delivery problems in supply chains with the objective of minimizing the overall scheduling and delivery cost. The scheduling criteria they consider include the sum of flowtimes, maximum lateness, and number of late jobs. For each problem considered, they either derive an efficient dynamic programming solution algorithm or show that it is intractable. Ji, He, and Cheng (2007) consider single-machine batch delivery scheduling to minimize the sum of the total weighted flow time and delivery cost. They show that the problem is $\mathcal{NP}$-hard in the strong sense and present polynomial-time algorithms for two special cases of the problem. Steiner and Zhang (2009) study single-machine batch delivery scheduling with batch setup times, where the objective is to minimize the sum of the weighted number of late jobs and delivery cost for multiple customers. They present a pseudo-polynomial dynamic programming algorithm for the restricted case where the tardy jobs are delivered separately at the end of the schedule. They then convert the algorithm into a fully polynomial-time approximation scheme (FPTAS) and show that it produces near-optimal solutions with low constant approximation ratios for the original problem. Steiner and Zhang (2011) consider single-machine batch delivery scheduling with due date assignment and batch setup times, where the objective is to minimize the sum of the weighted number of tardy jobs, due-date-assignment cost, and the batch delivery cost for a supplier whose system can be modeled by a single machine. They develop a pseudo-polynomial dynamic programming solution algorithm and then convert it into an FPTAS. Extending the problem studied by Steiner and Zhang (2011) to the case with multiple customers, Rasti-Barzoki and Hejazi (2013) formulate it as an integer programming problem, and present a heuristic and a branch-and-bound method to solve it. Yin, Cheng, Hsu, and Wu (2013) consider single-machine batch delivery scheduling with an assignable common due window. The objective is to find the optimal size and location of the window, the optimal dispatch date for each job, and an optimal job sequence to minimize a cost function based on earliness, tardiness, holding time, window location, window size, and batch delivery. They show that the problem is polynomially solvable by a dynamic programming algorithm. For recent results on batch delivery scheduling, the reader may refer to the survey paper of Chen (2010).

Perez-Gonzalez and Framinan (2014) review multi-agent scheduling research. In the last decade, multi-agent scheduling models has been successfully applied to many areas such as manufacturing, supply chain management, telecommunication services, project scheduling, and so on. Agnetis et al. (2004) and Baker and Smith (2003) first consider scheduling models in which two agents compete for the usage of a shared processing resource and each agent has its own objective function to optimize. The problem is either to find a schedule that minimizes a combination of the agents' objective functions or to find a schedule that satisfies each agent's requirements for its own objective function. Baker and Smith (2003) focus on finding a schedule that minimizes a combination of the agents' objective functions, which include the maximum of some regular functions (associated with each job), total (weighted) completion time, and number of late jobs. Agnetis et al. (2004) consider the same objective functions but focus on finding a schedule that satisfies each agent's requirements for its own objective function in various machine settings. Recently, some researchers consider multi-agent scheduling in the context of batch scheduling. Mor and Mosheiov (2011) study two-agent scheduling on a serial-batching machine. The objective is to minimize the flowtime of one agent, subject to an upper bound on the flowtime of the other agent, where they assume that the jobs and (agent-dependent) setup times are identical, and that the batches of the other agent must be processed continuously. They develop an $O(n^{3/2})$ algorithm to solve the problem. Li and Yuan (2012) consider two-agent scheduling on a common unbounded parallel-batching machine (i.e., jobs are processed in parallel), in which the job families are incompatible (i.e., the jobs from different families cannot be processed in the same batch) or compatible (i.e., all the jobs can be processed in the same batch). The objective is to minimize the objective function of one agent, while keeping the objective function value of the other agent from exceeding a given value. They provide plynomial-time and pseudo-polynomial-time algorithms to solve problems involving various combinations of