Computers & Industrial Engineering 111 (2017) 458–462

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Short Communication

A further study on two-agent scheduling on an unbounded serial-batch machine with batch delivery cost

Xianglai Qi, Jinjiang Yuan *

School of Mathematics and Statistics, Zhengzhou University, Zhengzhou, Henan 450001, People's Republic of China

ARTICLE INFO

Article history:
Received 9 May 2017
Received in revised form 17 July 2017
Accepted 19 July 2017
Available online 21 July 2017

Keywords:
Two-agent scheduling
Serial batch
Batch delivery
Pareto optimization

ABSTRACT

For the two-agent scheduling on an unbounded serial-batch machine with batch delivery cost, Yin et al. (2016) presented a comprehensive study, where the objective of each agent (A or B) is calculated by his scheduling cost plus his batch delivery cost proportional to the number of batches of this agent. Among their results, they provided a polynomial-time algorithm for minimizing the objective of agent A subject to the constraint that the objective of agent B does not exceed a given threshold value, where the criterion of agent A is the total completion time plus batch delivery cost and the criterion of agent B is the maximum lateness plus batch delivery cost. We show in this paper that their algorithm is incorrect by a counterexample and the algorithm presented in Kovalyov et al. (2015) for solving the same problem without batch delivery cost can be used to solve the problem in Yin et al. (2016) in polynomial time. We further study two corresponding Pareto scheduling problems and provide polynomial-time algorithms.

1. Introduction

Two-agent scheduling was first introduced in Baker and Smith (2003) and Agnetis, Mirchandani, Pacciarelli, and Pacifici (2004), and now has developed into a hot topic in scheduling research. Recently, two-agent scheduling on batch machines are studied in Li and Yuan (2012), Fan, Cheng, and Li (2013), Kovalyov, Soukhal, and Oulamara (2015), and Yin, Wang, Cheng, Wang, and Wu (2016).

We consider the two-agent scheduling on an unbounded serial-batch machine with batch delivery cost, which was first introduced and studied in Yin et al. (2016). In this scheduling model, two competing agents A and B will process their jobs on an unbounded serial-batch machine. For each $G \in \{A, B\}$, we use $\mathcal{J}^G = \{J_1^G, J_2^G, \ldots, J_{n_G}^G\}$ to denote the set of jobs of agent G, and moreover, the jobs in $\mathcal{J}^G$ are called the G-jobs. All jobs in $\mathcal{J}^A \cup \mathcal{J}^B$ are available for scheduling from time zero onwards. Each job J_j^G ($G \in \{A, B\}$) has a processing time p_j^G and a due date d_j^G. Let $n = n_A + n_B$ denote the total number of jobs. The machine used for processing the n jobs is an unbounded serial-batch machine. This means that the jobs are processed in batches and each batch can contain arbitrary number of jobs only of a common agent, where the processing time of a batch is given by the total processing time of the jobs in the batch and the jobs of the same batch complete simultaneously at the completion time of the batch. Whence a batch is completed in the machine it is delivered immediately and a delivery cost will be payed. For $G \in \{A, B\}$, a batch which just contains some G-jobs is called a G-batch. Each G-batch is associated with a sequence-independent batch setup time s_G and batch delivery cost φ_G independent of the number of jobs in the batch. All numerical data are assumed to be nonnegative integers. For simplicity, we assume the number of vehicles is sufficiently large and the delivery of batch is instantaneous.

Given a schedule σ, for each job J_j^G, where $G \in \{A, B\}$ and $j \in \{1, 2, \ldots, n_G\}$, we use $C_j^G(\sigma)$ to denote the completion time of job J_j^G and use $L_j^G(\sigma) = C_j^G(\sigma) - d_j^G$ to denote the lateness of job J_j^G. Let $f^G(\sigma)$ be the scheduling cost of agent G of schedule σ and $m_G(\sigma)$ the number of batches of the G-jobs in schedule σ. When no confusion can occur, we simply write $C_j^G(\sigma)$, $L_j^G(\sigma)$, $f^G(\sigma)$ and $m_G(\sigma)$ as C_j^G, L_j^G, f^G and m_G, respectively, by omitting σ. Then the objective function of agent G to be minimized is given by $f^G + m_G\varphi_G$.

For the bi-criteria scheduling on an unbounded serial-batch machine to minimize two objective functions f and g, we use the following notations for the expression of problems related to our research.

* Corresponding author.
E-mail addresses: qxl@zzu.edu.cn (X. Qi), yuanjj@zzu.edu.cn (J. Yuan).

http://dx.doi.org/10.1016/j.cie.2017.07.029
0360-8352/

- CP(U) problem $1|\text{s-batch}|f : g \leqslant U$. This is the constrained scheduling problem which aims at finding a schedule σ so that $f(\sigma)$ is minimized subject to the constraint condition $g(\sigma) \leqslant U$, where U is a threshold value of g.
- PP problem $1|\text{s-batch}|^{\#}(f, g)$. This is the Pareto scheduling problem which aims at finding all the Pareto points for minimizing f and g and for each Pareto point a corresponding Pareto optimal schedule.

Related to this paper is the work in Yin et al. (2016), in which the authors presented a comprehensive study for the CP(U) problems $1|\text{s-batch}|f^A + m_A\varphi_A : f^B + m_B\varphi_B \leqslant U$, where f^A and f^B are some regular scheduling criteria. Among the achievements in Yin et al. (2016), the authors presented a polynomial-time algorithm, called Sum-Max-DP Algorithm (shortly, algorithm SMDP), for problem $1|\text{s-batch}|\sum C_j^A + m_A\varphi_A : L_{\max}^B + m_B\varphi_B \leqslant U$.

When the delivery cost is not considered, i.e., $\varphi_A = \varphi_B = 0$, the CP($U$) problems $1|\text{s-batch}|f^A + m_A\varphi_A : f^B + m_B\varphi_B \leqslant U$ degenerate into $1|\text{s-batch}|f^A : f^B \leqslant U$, which were comprehensively studied in Kovalyov et al. (2015). Especially, Kovalyov et al. (2015) presented an $O(nn_A^2n_B^2)$-time algorithm for problem $1|\text{s-batch}|\sum C_j^A : L_{\max}^B \leqslant U$ by enumerating all possible choices of m_A and m_B.

By a counterexample, we show in Section 2 that the algorithm SMDP presented in Yin et al. (2016) for problem $1|\text{s-batch}|\sum C_j^A + m_A\varphi_A : L_{\max}^B + m_B\varphi_B \leqslant U$ is incorrect. Then we show that the algorithm presented in Kovalyov et al. (2015) for problem $1|\text{s-batch}|\sum C_j^A : L_{\max}^B \leqslant U$ can be used to solve problem $1|\text{s-batch}|\sum C_j^A + m_A\varphi_A : L_{\max}^B + m_B\varphi_B \leqslant U$ in $O(nn_A^2n_B^3)$ time. In Section 3, we further study two Pareto scheduling problems $1|\text{s-batch}|^{\#}(\sum C_j^A, L_{\max}^B)$ and $1|\text{s-batch}|^{\#}(\sum C_j^A + m_A\varphi_A, L_{\max}^B + m_B\varphi_B)$, which are related to the research in Section 2. By guessing and enumerating the possible maximum lateness of agent B, we show that the two Pareto scheduling problems are solvable in $O(nn_A^4n_B^5)$ time.

2. The constrained optimization

The algorithm SMDP in Yin et al. (2016) can be described as follows.

SMDP: For problem $1|\text{s-batch}|\sum C_j^A + m_A\varphi_A : L_{\max}^B + m_B\varphi_B \leqslant U$.

- Input: n_A, n_B, U, φ_A, φ_B, $p^A = \{p_1^A, p_2^A, \ldots, p_{n_A}^A\}$, $p^B = \{p_1^B, p_2^B, \ldots, p_{n_B}^B\}$, and $d^B = \{d_1^B, d_2^B, \ldots, d_{n_B}^B\}$.
- Step 1. Re-index the A-jobs in the SPT order and B-jobs in the EDD order.
- Step 2. [*Preprocessing*] Set $S(0,0) = \{(0,0,0,0,0)\}$ and $S(j_A, j_B) = \emptyset$ for $j_A = 0, 1, \ldots, n_A, j_B = 0, 1, \ldots, n_B$ with $j_A + j_B \geqslant 1$.
- Step 3. [*Generation*]
  For $j_A = 0$ to n_A do
  - For $j_B = 0$ to n_B do
    - For each $(F_A, m_A, m_B, q, x) \in S(j_A, j_B)$ do
      - **Case 1:** If $j_A + 1 \leqslant n_A$ and $x = 1$, then set $S(j_A+1, j_B) \leftarrow S(j_A+1, j_B) \cup \{(F_A', m_A, m_B, q, x)\}$, where $F_A' = F_A + (n_A + j_B - q)p_{j_A+1}^A$;
      - **Case 2:** If $j_A + 1 \leqslant n_A$, then set $S(j_A+1, j_B) \leftarrow S(j_A+1, j_B) \cup \{(F_A', m_A+1, m_B, j_A+j_B, 1)\}$, where $F_A' = F_A + (n_A - j_A)(s_A + p_{j_A+1}^A) + \varphi_A$;
      - **Case 3:** If $j_B + 1 \leqslant n_B, x = 2$ and $P(j_A, j_B + 1) - d_{j_A+j_B-q+1}^B + m_As_A + m_B(s_B + \varphi_B) \leqslant U$, then set $S(j_A, j_B+1) \leftarrow S(j_A, j_B+1) \cup \{(F_A', m_A, m_B, q, x)\}$, where $F_A' = F_A + (n_A - j_A)p_{j_B+1}^B$;
      - **Case 4:** If $j_B + 1 \leqslant n_B$ and $P(j_A, j_B + 1) - d_{j_B+1}^B + m_As_A + (m_B+1)(s_B + \varphi_B) \leqslant U$, then set $S(j_A, j_B+1) \leftarrow S(j_A, j_B+1) \cup \{(F_A', m_A, m_B+1, j_A+j_B, 2)\}$, where $F_A' = F_A + (n_A - j_A)(s_B + p_{j_B+1}^B)$;
    - Endfor
      [*Elimination*]
    - For any two states (F_A, m_A, m_B, q, x) and (F_A', m_A', m_B', q, x) in $S(j_A+1, j_B)$ with $F_A \leqslant F_A', m_A \leqslant m_A'$ and $m_B \leqslant m_B'$, eliminate the second one from $S(j_A+1, j_B)$.
    - For any two states (F_A, m_A, m_B, q, x) and (F_A', m_A', m_B', q, x) in $S(j_A, j_B+1)$ with $F_A \leqslant F_A', m_A \leqslant m_A'$ and $m_B \leqslant m_B'$, eliminate the second one from $S(j_A, j_B+1)$.
    - Endfor
  - Endfor

  Endfor
- Step 4. [*Result*] If $S(n_A, n_B) = \emptyset$, then report that the instance is infeasible. Otherwise, the optimal solution value is given by $F_A^* = \min\{F_A | (F_A, m_A, m_B, q, x) \in S(n_A, n_B)\}$ and the optimal solution can be found by backtracking.

However, in the processing of Case 4 of each iteration, to ensure that the resulted schedule meets the upper bound of the objective function value of agent B, the algorithm only checks whether the objective value of the current B-batch is no more than U or not. Since in Case 4, the number of B-batch increases by one, the objective value of previous B-batch should increase by φ_B accordingly. This can lead to the upper bound of the objective function value of agent B being not satisfied for pervious B-batch. In the following, we present a counterexample to further show the invalidity of the algorithm SMDP.

A counterexample: In the instance, we have $n_A = 1$ and $n_B = 2, s_A = \varphi_A = s_B = d_1^B = 0, \varphi_B = p_1^A = M$ (a sufficient large number), $p_j^B = 1$ ($j = 1, 2$), $d_2^B = 3M$, and $U = M + 2$.

For this instance, the above algorithm SMDP generates a schedule σ which schedules J_1^B in the time interval $[0, 1]$, J_1^A in the time interval $[1, M+1]$, and J_2^B in the time interval $[M+1, M+2]$. Then $L_{\max}^B(\sigma) = 1$. Since $\varphi_B = M$ and the B-jobs are partitioned into two batches in σ, i.e., $m_B(\sigma) = 2$, it is not hard to verify that $L_{\max}^B(\sigma) + m_B(\sigma)\varphi_B = 2M + 1 > U$. Thus, the schedule σ generated by algorithm SMDP is even not feasible.

In fact, in an optimal schedule, to ensure the objective value of agent B does not exceed U, the two jobs of agent B have to be assigned in one batch and processed in the time interval $[0, 2]$, and job J_1^A is scheduled at last in the time interval $[2, M+2]$. Such an optimal schedule, denoted σ^*, has $L_{\max}^B(\sigma^*) + m_B(\sigma^*)\varphi_B = M + 2 = U$ and $\sum C_j^A(\sigma^*) + m_A(\sigma^*)\varphi_A = M + 2$. This means that the algorithm SMDP presented in Yin et al. (2016) for problem $1|\text{s-batch}|\sum C_j^A + m_A\varphi_A : L_{\max}^B + m_B\varphi_B \leqslant U$ is invalid.

To amend the above flaw, we will show that the algorithm presented in Kovalyov et al. (2015) for problem $1|\text{s-batch}|\sum C_j^A : L_{\max}^B \leqslant U$ can be used to solve problem $1|\text{s-batch}|\sum C_j^A + m_A\varphi_A : L_{\max}^B + m_B\varphi_B \leqslant U$ in $O(nn_A^2n_B^3)$ time. We first introduce an auxiliary problem.

- RCP($U; b_A, b_B$) problem $1|\text{s-batch}, (b_A, b_B)|\sum C_j^A : L_{\max}^B \leqslant U$. This is the restricted version of the CP(U) problem $1|\text{s-batch}|\sum C_j^A : L_{\max}^B \leqslant U$, in which we require that each feasible schedule has just b_A A-batches and b_B B-batches, i.e., $m_A(\sigma) = b_A$ and $m_B(\sigma) = b_B$ for every feasible schedule σ, where $1 \leqslant b_A \leqslant n_A$ and $1 \leqslant b_B \leqslant n_B$. Moreover, we use $F(U; b_A, b_B)$ to denote the opti-