Decision Making in Manufacturing and Services
Vol. 1 • 2007 • No. 1–2 • pp. 25–34

Batch Scheduling of Deteriorating Products

Maksim S. Barketau*, T.C. Edwin Cheng**,
Mikhail Y. Kovalyov***, C.T. Daniel Ng ****

Abstract. In this paper we consider the problem of scheduling N jobs on a single machine, where the jobs are processed in batches and the processing time of each job is a simple linear increasing function depending on job's waiting time, which is the time between the start of the processing of the batch to which the job belongs and the start of the processing of the job. Each batch starts from the setup time S. Jobs which are assigned to the batch are being prepared for the processing during time $S_0 < S$. After this preparation they are ready to be processed one by one. The non-negative number b_i is associated with job i. The processing time of the i-th job is equal to $b_i(s_i - (s_i^b + S_0))$, where s_i^b and s_i are the starting time of the b-th batch to which the i-th job belongs and the starting time of this job, respectively. The objective is to minimize the completion time of the last job. We show that the problem is NP-hard. After that we present an $O(N)$ time algorithm solving the problem optimally for the case $b_i = b$. We further present an $O(N^2)$ time approximation algorithm with a performance guarantee 2.

Keywords: scheduling, batching, remanufacturing, deterioration.

Mathematics Subject Classification: 90B35.

Received/Revised: 07 May 2007/02 June 2007

1. INTRODUCTION

We consider a situation of a distant communication involving radio, mobile or internet channel in the presence of overloaded communication network which is being disturbed by outside noises. The goal is to transfer a number of data packages from the client to the server. Starting a transfer is preceded by the establishing of a connection between client and server, which takes a period of time. After the connection has been

* Faculty of Economics, Belarus State University, Belarus; Department of Logistics, The Hong Kong Polytechnic University, China; United Institute of Informatics Problems, National Academy of Sciences of Belarus, Belarus. E-mail: barketau@mail.ru.
** Department of Logistics, The Hong Kong Polytechnic University, China. E-mail: Edwin.Cheng@inet.polyu.edu.hk.
*** Faculty of Economics, Belarus State University, Belarus. E-mail: koval@newman.bas-net.by.
**** Department of Logistics, The Hong Kong Polytechnic University, China. E-mail: Daniel.Ng@inter.polyu.hk.

established the communication starts. Due to the overloaded communication network and outside noises errors happen while transferring data, which implies prolonged time of the transfer. The longer the connection is being used the more errors happen. Thus, the transfer time of a data package depends on the time which data package awaits for the transfer and increases linearly with the time elapsed since the establishing of the connection. Another option is to establish new connection and transfer data packages through a new channel. The problem is to minimize the total time spent on the establishing connections and transfer of all data packages. The described situation can be observed while connecting to the overloaded internet site or while performing mobile communication session from the places with the geographical location that influence the quality of a mobile communication.

Consider another example where a company sequentially performs a set of tasks concerned with the improvement of its office or production facilities. The company can combine several tasks into a single package and hire a group of workers to perform all the tasks assigned to this package. On receiving the package, the group of workers spends a fixed time to prepare for their work. After preparation the workers start to perform the tasks at a normal pace. However, with the passage of time, they become tired and their pace decreases, which depends on the nature of the tasks. The objective is to combine the tasks into packages and to schedule the packages so as to minimize the completion time of the last task.

In this paper we study a scheduling model for making a decision in the above situations. In the next section we formulate the problem formally.

2. PROBLEM DESCRIPTION

Consider set $I = \{1, 2, \ldots, N\}$ of N jobs that are to be scheduled on a single machine. The jobs are processed in batches. Each batch starts from the setup time S. The jobs that are assigned to the batch are prepared for the processing during time $S_0 < S$. After this they are ready to be processed one by one. The processing time of a job is a simple linear increasing function of the job's waiting time. Let s_i^b and s_i be the starting time of the b-th batch to which the i-th job belongs and the starting time of this job, respectively. The processing time of the i-th job is equal to $b_i(s_i - (s_i^b + S_0))$. Thus, the waiting time of the first job assigned to the batch is $S - S_0$. The problem is to minimize the makespan of the schedule. We denote this problem by $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$.

Deterioration of jobs awaiting processing is a situation received increased attention of the researchers recently. Deterioration causes increased processing time of the job. First articles devoted to problems of deteriorating jobs scheduling are works of Mosheiov (1994), Mosheiov (1992) and Browne and Yechiali (1990). A number of works among which the work of Cheng and Ding (2001) has been published recently. For the overview of scheduling problems with jobs processing times dependant on the starting time of job processing, see Cheng, Ding and Lin (2004).

We introduced the aspect of batching in the above mentioned models. In our model jobs are processed in batches and job processing time depends on the time elapsed since the start of batch processing.

In the next section we introduce reader to some preliminary facts about the studied problem.

3. PRELIMINARIES

Consider the schedule which consists of k batches. Let jobs of the j-th batch comprise set K_j. The processing time of the j-th batch without the setup time then can be calculated as $T(j) = (S-S_0)(\prod_{i\in K_j}(b_i+1)-1)$. Thus, the makespan of the schedule is calculated as follows.

$$C_{max} = \sum_{j=1}^{k}(S + (S - S_0)(\prod_{i\in K_j}(b_i + 1) - 1)) = kS_0 + (S - S_0)\sum_{j=1}^{k}\prod_{i\in K_j}(b_i + 1). \quad (1)$$

Note that taking into account the inequality between geometric and arithmetic average described in Beckenbach and Bellman (1971), the lower bound of the C_{max} is

$$C_{max} = kS_0 + (S - S_0)\sum_{j=1}^{k}\prod_{i\in K_j}(b_i + 1) \geq kS_0 + (S - S_0)k\sqrt[k]{\prod_{i=1}^{N}(b_i + 1)}. \quad (2)$$

Moreover, this inequality turns to equality if and only if $\prod_{i\in K_j}(b_i + 1) = \sqrt[k]{\prod_{i=1}^{N}(b_i + 1)}, j = 1, \dots k$.

4. COMPLEXITY RESULTS

As pointed out by Cheng et al. (2004) the basic problem that is widely used to prove the NP-hardness of scheduling problems with lineary deteriorating jobs is Subset Product problem described in Garey and Johnson (1979). We will use the following natural modification of Subset Product problem, which is proved to be strongly NP-Complete in Barketau et al. (2007).

Problem Product Partition: There are N positive integers $u_1, u_2, \dots, u_N$. The question is whether there is a subset $X \subset I := \{1, 2, \dots, N\}$ such that $\prod_{i\in X} u_i = \prod_{i\in I\setminus X} u_i$.

Knowing the intractability of the problem Product Partition, we use it to establish the intractability of problem $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$. But first formulate the corresponding recognition problem which we also denote by $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$.

Problem $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$: The set $I = \{1, 2, \dots, N\}$ of N jobs is to be scheduled on a single machine. The positive number b_i is associated with

each job i. The jobs are processed in batches. Each batch starts with a setup time S. All jobs assigned to the batch are prepared for the processing during time S_0. After this period of time jobs of the batch are ready to be processed one by one. Thus, the waiting time of the first job is equal to $S - S_0$. The processing time of a job is a simple linear increasing function that depends on the job's waiting time. Let s_i^b and s_i be the starting time of the b-th batch to which the i-th job belongs and the starting time of this job, respectively. The processing time of the i-th job is equal to $b_i(s_i - (s_i^b + S_0))$. The question is if there is a schedule with a processing time less or equal to C.

Theorem 1. *Problem* $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$ *is NP-hard.*

Proof. Given the instance of problem PRODUCT PARTITION construct the following instance of problem $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$: There are N jobs. Let $b_i = u_i - 1, 1 \le i \le N$. Let $S_0 = P - 2\lfloor\sqrt{P}\rfloor - 1$, where $P = \prod_{i\in I} u_i = \prod_{i=1}^{N}(b_i + 1)$. Let $S = S_0 + 1$. Let $C = 2P - 2\lfloor\sqrt{P}\rfloor - 2$. We show that there is a required subset for problem PRODUCT PARTITION if and only if there is a schedule for the problem $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$ with C_{max} not greater than $C = 2P - 2\lfloor\sqrt{P}\rfloor - 2$. Note that our reduction is not pseudopolynomial because of the presence of large numbers P and $\sqrt{P}$ in the input. But this reduction can be made in time polynomial of the length of the input of problem PRODUCT PARTITION.

Assume that there is subset X such that $\prod_{i\in X} u_i = \prod_{i\in I\setminus X} u_i = \sqrt{P}$. Consider a schedule that consists of two batches. First batch comprises exactly those jobs whose coefficients b_i were counted on the basis of $u_i, i \in X$. Second batch comprises the remaining jobs. Count C_{max} of this schedule using (1): $C_{max} = 2S_0 + \sum_{j=1}^{2}\prod_{i\in K_j}(b_i + 1) = 2(P - 2\lfloor\sqrt{P}\rfloor - 1) + \prod_{i\in X} u_i + \prod_{i\in I\setminus X} u_i = 2P - 2\lfloor\sqrt{P}\rfloor - 2 = C$.

Assume now that there is a schedule with $C_{max} \le C$. Denote by K_j the set of jobs in the j-th batch. If there is only one batch in the schedule, then its makespan is $C_{max} = S_0 + \prod_{i=1}^{N}(b_i+1) = S_0 + P = 2P - 2\lfloor\sqrt{P}\rfloor - 1 > C$. Thus, the schedule contains more then one batch. If there are $k > 2$ batches in the schedule, then, according to (2), the lower bound on the makespan is: $C_{max} \ge kS_0 + k\sqrt[k]{\prod_{i=1}^{N}(b_i + 1)} = kS_0 + k\sqrt[k]{P} \ge kS_0 \ge 2P - 2\sqrt{P} - 2 + P - 4\sqrt{P} - 1 > C$ if $P > 25$. Thus the schedule contains exactly two batches. Calculate its makespan and lower bound: $C_{max} = 2S_0 + \sum_{j=1}^{2}\prod_{i\in K_j}(b_i + 1) \ge 2S_0 + 2\sqrt{\prod_{i=1}^{N}(b_i + 1)} = 2S_0 + 2\sqrt{P} = 2P - 4\lfloor\sqrt{P}\rfloor + 2\sqrt{P} - 2 \ge C$. Moreover, taking into account (2), C_{max} is equal to C if and only if $\prod_{i\in K_1}(b_i + 1) = \prod_{i\in K_2}(b_i + 1) = \sqrt{P}$. Therefore, if we compose X of exactly those $i \in I$ on the basis of which were calculated coefficients b_i of jobs from the first batch, X will be the required set of problem PRODUCT PARTITION. □

Note that it is possible to prove the strong NP-hardness of modified problem $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$, in which numbers S and S_0 in the input are coded in a special manner to allow large values. Thus, the following corollary takes place.

Corollary 1. *There is no algorithm polynomial of* N *and* $b_{max} = \max\{b_i | 1 \le i \le N\}$ *solving problem* $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$ *optimally.*

5. THE CASE OF CONSTANT DETERIORATION RATES

Because the general problem appeared to be NP-hard, in this section we consider the special case of the problem in which coefficients b_i are equal for all the jobs, i.e., $b_i = b$.

In this case the makespan of the schedule is determined by the number of batches in the schedule and the set of their sizes. Using formula (1) we find the following expression for the makespan of the schedule.

$$C_{max} = f_k(\{l_1, l_2, \ldots, l_k\}) = kS_0 + (S - S_0)\sum_{j=1}^{k}(b+1)^{l_j}, \tag{3}$$

where k is the number of batches in the schedule, and l_j is the number of jobs in the j-th batch.

In the following lemma we formulate the property of the optimal schedule containing k batches.

Lemma 1. *Let $N = kz + r$, where z and r are integers, and $z \geq 0, 0 \leq r < k$. Among the schedules made of k batches the optimal schedule contains r batches made of $z+1$ jobs each and $k - r$ batches made of z jobs each.*

Proof. Consider arbitrary feasible schedule containing k batches. Divide all the batches of this schedule into two following sets. Let set B_z and set B^{z+1} contain batches that comprise not more than z jobs and batches that comprise not less than $z + 1$ jobs, respectively.

Assume first, that $|B^{z+1}| > r$. In this case there is at least one batch of size $z-m, m > 0$, in set B_z. Select one batch of the minimum size $z+l, l > 0$, from the set B^{z+1} and transfer one job from this batch to the batch of size $z - m, m > 0$. Taking into account formula 3, the changes in the makespan of the schedule are calculated as follows.

$$\begin{aligned} \Delta C_{max} &= (S - S_0)((b+1)^{z-m+1} + (b+1)^{z+l-1} - (b+1)^{z-m} - (b+1)^{z+l}) = \\ &= (S - S_0)((b+1)^{z-m} b(1 - (b+1)^{l+m-1})) < 0. \end{aligned}$$

Because ΔC_{max} is negative, we can repeat this step to obtain the schedule in which $|B^{z+1}| = r$, and, consequently, $|B_z| = k - r$.

If $|B^{z+1}| < r$, than we use the same technique to construct the schedule with the smaller makespan and for which $|B^{z+1}| = r$.

On constructing the schedule with $|B^{z+1}| = r$, consider the sizes of batches in the set B^{z+1}. Assume that there is a batch with the size $z + l, l > 1$. In this case there is at least one batch in the set B_z with the size $z - m, m > 0$. Transfering the job from the batch of size $z + l$ to the batch of size $z - m$, we construct the schedule with the smaller makespan. Repeating this step, we obtain the schedule with the smaller makespan than the initial one and for which $|B^{z+1}| = r$, each batch in the set B^{z+1} have exactly $z + 1$ jobs and each batch in the set B_z have exactly z jobs. □

Note that to prove Lemma 1 one can use results of the article Al-Anzi et al. (2007).

Lemma 1 justifies the following algorithm for the special case under consideration.

Algorithm A

Step 1: For each $k = 1, \ldots, N$ do Step 2

Step 2: Let $N = kz_k + r_k$ where $z_k \geq 0$ and $0 \leq r_k < k$. Find the makespan of the optimal schedule made of k batches: $C^k_{max} = kS_0 + (S - S_0)(r_k(b+1)^{z_k+1} + (k - r_k)(b+1)^{z_k})$.

Step 3: Let $C^{k_0}_{max} = \min\{C^k_{max} | k = 1, \ldots, N\}$. Optimal schedule contains k_0 batches and have r_{k_0} batches made of $z_{k_0}+1$ jobs and $k - r_{k_0}$ batches made of z_{k_0} jobs.

Algorithm A runs in $O(N)$ time. If function C^k_{max} of k is unimodal than algorithm can be implemented to run in $O(\log N)$ time.

6. APPROXIMATION ALGORITHM

Let $\Pi(k, G)$ be the problem of scheduling the set of jobs $G \subseteq I = \{1, 2, \ldots, N\}$ optimally in k batches. We first present the approximation algorithm $\mathrm{B}(k, G)$, which builds the approximate schedule for problem $\Pi(k, G)$. Note that we don't restrict algorithm $\mathrm{B}(k, G)$ to construct schedules which contain exactly k batches. Instead algorithm $\mathrm{B}(k, G)$ may construct schedules containing more than k batches.

Let $P_G = \prod_{i \in G}(b_i + 1)$. Let $F(G, x) = \{i | i \in G \text{ and } b_i + 1 > x\}$.

Algorithm B(k, G)

Step 1: If $F(G, \sqrt[k]{P_G}) \neq \phi$ then assign each job of set $F(G, \sqrt[k]{P_G})$ into separate batch alone and close each of the resulting $|F(G, \sqrt[k]{P_G})|$ batches. Make call to the algorithm $\mathrm{B}(k - |F(G, \sqrt[k]{P_G})|, G \backslash F(G, \sqrt[k]{P_G}))$ and append batches of the resulting schedule to the existing ones. Stop.

If $F(G, \sqrt[k]{P_G}) = \phi$ then perform Steps 2 and 3.

Step 2: (assignment to basic batches) Now we assign jobs of set G to basic batches. We call the batches we assign jobs in during this step *basic* batches and batches which are composed on Step 3 *extra* batches. Let j be the current basic batch to which we assign jobs. We start from $j = 1$. Assign jobs one by one in arbitrary order to the batch j until $P_{K_j} > \sqrt[k]{P_G}$, where K_j is the set of jobs in batch j. As soon as after adding job, P_{K_j} becomes greater then $\sqrt[k]{P_G}$ close current batch j and set $j = j + 1$. Continue assigning jobs to basic batches.

Close the last basic batch. Assume there are t basic batches. Let $m = t + 1$.

Step 3: (assignment to extra batches) We start to move last jobs from the basic batches to the extra batches. Consider current extra batch m. For each basic batch j with $P_{K_j} > \sqrt[k]{P_G}$ take the last job l assigned to the batch and move this job to the extra batch m if $P_{K_m \cup \{l\}} \leq \sqrt[k]{P_G}$. If $P_{K_m \cup \{l\}} > \sqrt[k]{P_G}$ then assign job l to the batch $m + 1$ and let $m = m + 1$. Continue assigning last jobs from the basic batches to the extra batches on Step 3.

Note that the number of extra batches u is not greater than the number of basic batches t.

The computational complexity of algorithm $\mathrm{B}(k, G)$ is $O(|G|)$.

On Step 1 jobs from set $F(G, \sqrt[k]{P_G})$ are assigned to the batches. Each job of set $F(G, \sqrt[k]{P_G})$ is assigned to the separate batch which is closed after assignment. Problem $\Pi(k, G)$ is reduced to problem $\Pi(k - |F(G, \sqrt[k]{P_G})|, G\backslash F(G, \sqrt[k]{P_G}))$ and corresponding recursive call to algorithm $\mathrm{B}(k - |F(G, \sqrt[k]{P_G})|, G\backslash F(G, \sqrt[k]{P_G}))$ is made. Step 1 of algorithm $\mathrm{B}(k, G)$ is justified by the following Lemma.

Lemma 2. *There is an optimal schedule for problem $\Pi(k, G)$ in which every job from set $F(G, \sqrt[k]{P_G})$ comprises a separate batch in the schedule.*

Proof. Consider an optimal schedule for problem $\Pi(k, G)$. The makespan of this optimal schedule C_{max} is calculated according to (1).

$$C_{max} = kS_0 + (S - S_0)\sum_{j=1}^{k}\prod_{i\in K_j}(b_i + 1) = kS_0 + (S - S_0)\sum_{j=1}^{k} P_{K_j},$$

where $K_j, j = 1, \ldots, k$, is the set of jobs in the j-th batch of the optimal schedule.

Let i be an arbitrary job from set $F(G, \sqrt[k]{P_G})$. Assume that i is in the batch v, i.e., $i \in K_v$. According to the definition of set $F(G, \sqrt[k]{P_G})$, $b_i + 1 > \sqrt[k]{P_G}$ and thus, $P_{K_v} > \sqrt[k]{P_G}$.

There must be batch q for which $P_{K_q} < \sqrt[k]{P_G}$. Prove this. Assume that for every batch $j, j = 1, \ldots, k$, $P_{K_j} \geq \sqrt[k]{P_G}$. Additionally we have that $P_{K_v} > \sqrt[k]{P_G}$. Thus, the following is true. $P_G = \prod_{j=1}^{k} P_{K_j} > (\sqrt[k]{P_G})^k = P_G$. There is a contradiction. Thus, there is such batch q that $P_{K_q} < \sqrt[k]{P_G}$.

Assume that there are other jobs in batch v except job i. Move these jobs from batch v to batch q. Calculate the difference in the makespan of the resulting schedule C'_{max} and the makespan of the original schedule C_{max} divided by $S - S_0$ for the convenience of representation. We have

$$\frac{C'_{max} - C_{max}}{S - S_0} = P_{K'_v} + P_{K'_q} - P_{K_v} - P_{K_q},$$

where $K'_v = \{i\}$ and $K'_q = K_q \cup (K_v\backslash\{i\})$.

We further have

$$P_{K'_v} + P_{K'_q} - P_{K_v} - P_{K_q} = b_i + 1 + P_{K_q}P_{K_v\backslash\{i\}} - (b_i + 1)P_{K_v\backslash\{i\}} - P_{K_q} =$$

$$= P_{K_v\backslash\{i\}}(P_{K_q} - b_i - 1) - (P_{K_q} - b_i - 1) = (P_{K_v\backslash\{i\}} - 1)(P_{K_q} - b_i - 1) \leq 0$$

Thus, we obtained the optimal schedule in which job i alone comprises v-th batch. Repeating the same technique for the remaining jobs of the set $F(G, \sqrt[k]{P_G})$ we obtain the optimal schedule in which every job from set $F(G, \sqrt[k]{P_G})$ comprises a separate batch in the schedule. □

Let $F_0 := F(G, \sqrt[k]{P_G})$ and $H_0 := G\backslash F(G, \sqrt[k]{P_G})$. According to Lemma 2 $|F_0|$ batches from optimal schedule is known. After scheduling these batches on Step 1 algorithm $\mathrm{B}(k, G)$ makes a recursive call to algorithm $\mathrm{B}(k - |F_0|, H_0)$ to approximate the remaining $k - |F_0|$ batches. Note that before going to Step 2 algorithm $\mathrm{B}(k, G)$

makes a number of recursive calls to itself with changed parameters. Let there are s recursive calls on Step 1. Call $j, j = 1, \ldots, s$, is made to algorithm $\mathrm{B}(k_j, H_{j-1})$, where $k_j = k - |\cup_{i=0}^{j-1} F_i|$, $F_j = F(H_{j-1}, \sqrt[k_j]{P_{H_{j-1}}})$ and $H_j = H_{j-1} \backslash F_j$. During recursive call $j, j = 0, \ldots, s-1$, jobs from set F_j is scheduled to the separate batch each and these batches become closed. Thus, during recursive call s jobs from set $F = \cup_{j=0}^{s-1} F_j$ is already scheduled into $|F|$ batches. Applying Lemma 2 s times we obtain that there is an optimal schedule which contains jobs from set F comprising a separate batch each and the remaining jobs $G \backslash F$ scheduled into $k - |F|$ batches. We will consider only such optimal schedules in the remaining part of the section.

On Step 2 algorithm $\mathrm{B}(k, G)$ assign jobs to t basic batches. Note that $t \leq k$. Prove this. Assume that $t > k$. Let K'_j be the set of jobs in batch $j, j = 1, \ldots, t$, after Step 2 of algorithm $\mathrm{B}(k, G)$. According to the Step 2 of algorithm $\mathrm{B}(k, G)$ $P_{K'_j} > \sqrt[k]{P_G}, j = 1, \ldots, t-1$. Then we have $P_G = \prod_{j=1}^{t} P_{K'_j} \geq \prod_{j=1}^{k} P_{K'_j} > (\sqrt[k]{P_G})^k = P_G$. We obtained a contradiction. Thus, $t \leq k$.

The estimation of approximation guarantee of algorithm $\mathrm{B}(k, G)$ is given in the following lemma.

Lemma 3. *Let C_{max} and C^k_{max} be the makespan of an optimal schedule for problem $\Pi(k, G)$ and the makespan of the schedule built by algorithm $B(k, G)$, respectively. The approximation guarantee of algorithm $B(k, G)$ is $C^k_{max}/C_{max} \leq 2$.*

Proof. Let s be the number of recursive calls made on Step 1 of algorithm $\mathrm{B}(k, G)$. Let $F_j, j = 0, \ldots, s-1$ be set of jobs which are assigned to the separate batch each on Step 1 during j-th recursive call. Assume that $f = |F_0| + |F_1| + \ldots + |F_{s-1}|$. Thus, first f batches of the schedule built by algorithm $\mathrm{B}(k, G)$ are made of the single job each and these jobs are from the set $F = F_0 \cup F_1 \cup \ldots \cup F_{s-1}$. Let $H = G \backslash F$. Applying Lemma 2 s times we obtain that the optimal schedule for problem $\Pi(k, G)$ contains the same f batches made of the single job each where jobs are from the set F. The remaining $k - f$ batches in the optimal schedule are made of jobs from set H. Therefore, taking into account (2), the lowerbound on the makespan of the optimal schedule is calculated as follows:

$$C_{max} \geq kS_0 + (S - S_0) \sum_{i \in F} (b_i + 1) + (S - S_0)(k - f) \sqrt[k-f]{P_H}.$$

Find now the upper bound on the makespan of the schedule built by algorithm $\mathrm{B}(k, G)$. As we pointed out first f batches of this schedule are made of single job each where jobs are from set F. After s-th recursive call to algorithm $\mathrm{B}(k - f, H)$ the remaining part of the schedule is created on Step 2 and 3 of the algorithm from the jobs of set H. Thus, batches $j, f + 1 \leq j \leq f + t$, are basic and batches $j, f + t + 1 \leq j \leq f + t + u$, are extra batches. Note that according to the construction of batches on Steps 2 and 3 of algorithm $\mathrm{B}(k - f, H)$ after Step 3 for each basic and extra batch $j, f + 1 \leq j \leq f + t + u$, we have $P_{K_j} \leq \sqrt[k-f]{P_H}$, where K_j is the set of jobs in the j-th batch. Because the number of basic batches t is not greater than $k - f$ and

the number of extra batches u is not greater than t the following expression is an upperbound on the makespan of the shedule.

$$C^k_{max} \leq fS_0 + (S - S_0)\sum_{i\in F}(b_i + 1) + 2(k - f)S_0 + 2(S - S_0)(k - f)\sqrt[k-f]{P_H} \leq$$

$$\leq 2kS_0 + (S - S_0)\sum_{i\in F}(b_i + 1) + 2(S - S_0)(k - f)\sqrt[k-f]{P_H}.$$

Using the lowerbound on the makespan of the optimal schedule and the upperbound on the makespan of the schedule built by algorithm $\mathrm{B}(k, G)$ calculate C^k_{max}/C_{max}.

$$\frac{C^k_{max}}{C_{max}} \leq \frac{2k\frac{S_0}{S-S_0} + \sum_{i\in F}(b_i + 1) + 2(k - f)\sqrt[k-f]{P_H}}{k\frac{S_0}{S-S_0} + \sum_{i\in F}(b_i + 1) + (k - f)\sqrt[k-f]{P_H}} \leq 2$$

□

Now we present the approximation algorithm for problem $1|S, p_i = b_i(s_i - (s^b_i + S_0))|C_{max}$. The idea of the algorithm is to apply algorithm $\mathrm{B}(k, I)$ for $k = 1, \ldots, N$, and choose the schedule with the minimal makespan.

Algorithm *C*

Step 1: For each $k = 1, \ldots, N$ make call to algorithm $\mathrm{B}(k, I)$ and obtain a schedule with makespan C^k_{max}.

Step 2: Find $C^0_{max} = \min\{C^k_{max} | k = 1, \ldots, N\}$. The corresponding schedule is the output of the algorithm *C*.

The computational complexity of algorithm *C is* $O(N^2)$.

Theorem 2. *Let* C_{max} *and* C^0_{max} *be the makespan of an optimal schedule for problem* $1|S, p_i = b_i(s_i - (s^b_i + S_0))|C_{max}$ *and the makespan of the schedule built by algorithm C, respectively. Algorithm C runs in* $O(N^2)$ *time and its approximation guarantee is* $C^0_{max}/C_{max} \leq 2$.

Proof. Follows from Lemmas 2 and 3. □

7. CONCLUSIONS

In this paper we studied a problem with elements of batch scheduling and simple deterioration. Specifically, jobs are processed in batches preceded by a setup time and the processing time of a job depends on the time elapsed since the start of the batch that the job belongs to. A non-negative numbers b_i is associated with job i. The processing time of the i-th job is equal to $b_i(s_i - (s^b_i + S_0))$, where s^b_i, s_i and S_0 are the starting time of the b-th batch to which the i-th job belongs, the starting time of this job and batch preparation time, respectively.

We have shown that the problem is NP-hard.

For the special case of the problem with $b_i = b$ we presented an $O(N)$ time algorithm solving the problem optimally. For the general case we presented an $O(N^2)$ time approximation algorithm with a performance guarantee 2.

Further research can be undertaken in the direction of developing approximation algorithms with a good asymptotic behavior for the general problem $1|S, p_i = b_i(s_i - (s_i^b + S_0))|C_{max}$.

REFERENCES

1. Al-Anzi F.S., Allahverdi A., Kovalyov M.Y., *Batching Deteriorating Items with Applications in Computer Communication and Reverse Logistics,* European Journal of Operational Research, accepted 2007.

2. Barketau M.S., Cheng T.C.E., Kovalyov M.Y., Ng C.T.D., *Computational complexity of "Product Partition" and related problems,* working paper 2007.

3. Beckenbach E.F., Bellman R., *Inequalities,* Springer-Verlag, New-York 1971.

4. Browne S., Yechiali U., *Schedulinng deteriorating jobs on a single processor,* Operations Research, 38 (1990), 495–498.

5. Cheng T.C.E., Ding Q., *Single machine scheduling with step-deterioration processing times,* European Journal of Operational Research, 134 (2001), 623–630.

6. Cheng T.C.E., Ding Q., Lin B.M.T., *A concise survey of scheduling with time-dependent processing times,* European Journal of Operational Research, 152 (2004), 1–13.

7. Garey M.R., Johnson D.S., *Computers and intractability: A guide to the Theory of NP-Completeness.* Freeman, New York 1979.

8. Mosheiov G., *V-Shaped policies for scheduling deteriorating jobs,* Operations Research, 39 (1992), 6, 979–991.

9. Mosheiov G., *Scheduling jobs under simple linear deterioration,* Computers and Operations Research, 21 (1994), 6, 653–659.