Batch Scheduling of Deteriorating Reworkables

M.S. Barketau[a,b], T.C.E. Cheng[b,*], M.Y. Kovalyov[a,c]

[a] *Faculty of Economics, Belarusian State University, Nezavisimosti 4, 220050 Minsk, Belarus*

[b] *Department of Logistics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*

[c] *United Institute of Informatics Problems, National Academy of Sciences of Belarus, Surganova 6, 220012 Minsk, Belarus*

Abstract

The problem of scheduling the production of new and recoverable defective items of the same product manufactured on the same facility is studied. Items are processed in batches. Each batch comprises two sub-batches processed consecutively. In the first sub-batch, all the items are newly manufactured. Some of them are of the required good quality and some are defective. The defective items are remanufactured in the second sub-batch. They deteriorate while waiting for rework. This results in increased time and cost for their remanufacturing. All the items in the same sub-batch complete at the same time, which is the completion time of the last item in the sub-batch. Each remanufactured defective item is of the required good quality. It is assumed that the percentage of defective items in each batch is the same. A setup time is required to start batch processing and to switch from manufacturing to remanufacturing. The demands for good quality items over time are given. The objective is to find batch sizes such that the total setup and inventory holding cost is minimized and all the demands are satisfied. Dynamic programming algorithms are presented for the general problem and some important special cases.

Key Words: batching, scheduling, remanufacturing, deterioration.

*Corresponding author

1 Introduction

Remanufacturing of defective products has increasingly become a popular approach to manufacturing lately. Such a practice allows the manufacturer to extract value from defective products, reduce disposal costs, and comply with environmental legislation. Many factors such as an unstable production environment, workers' mistakes, and usage of inappropriate manufacturing facilities can lead to the production of defective products.

The production of industrial steel ball bearings is a real-life example that demonstrates the practice of remanufacturing. The first step of ball bearings production is to pour molten steel into a mould, each producing a batch of the bearings. Due to varying compositions of the steel and inherent instability of the production facility, internal cavities may occasionally be created in some of the manufactured ball bearings, which render them defective. Ball bearings having such cavities are considered to be defective and are to be reworked. Thus, the second step is to reheat and remanufacture each defective ball bearing produced in the first step. The longer a ball bearing waits for the rework, the cooler it becomes. Consequently, the time needed to reheat a ball bearing increases with the time elapsed since its remanufacturing. In order to improve production efficiency, ball bearings are produced in batches and the defective units of each batch are reworked in a batch on the same facility immediately after they have been detected.

In the following section we propose a scheduling model to study the described production process characterized by manufacturing and remanufacturing of identical items in batches on a single facility, and make a brief review of the related literature.

2 Model description

We consider a situation in which N identical good quality items of a single product are to be manufactured on a single machine. Given a schedule, let us consider the corresponding sequence of item completion times. We say that an item is in *position* j if its completion time is jth in the above sequence. All the items are available for processing at time zero. *Positional deadlines* $d_1, d_2, \ldots, d_N$ are given such that an item in position j should be completed by d_j, $j = 1, \ldots, N$. We can assume without loss of generality that $d_1 \leq \cdots \leq d_N$.

The information on $d_1, \ldots, d_N$ is obtained from the manufacturer's production planning system. There is the following correspondence between the positional deadlines and demands for good quality items over time. Let $Q_1, \ldots, Q_u$ be the quantities of good quality items demanded at times $t_1, \ldots, t_u$, respectively, $0 < t_1 < \cdots < t_u$. The equivalent positional deadline formulation can be obtained by setting $N = \sum_{j=1}^{u} Q_j$ and $d_j = t_1$, $j = 1, \ldots, Q_1$, $d_j = t_2$, $j = Q_1 + 1, \ldots, Q_1 + Q_2$, $\ldots$, $d_j = t_u$, $j = N - Q_u + 1, \ldots, N$.

Scheduling problems with positional due dates, also known as generalized due dates, were studied, among others, by Hall, Sethi and Sriskandarajah [5], Gordon and Kubiak [3], Tanaka and Vlach [10] and Qi, Yu and Bard [9].

The manufacturing of a good quality item requires one or two operations that we call *work* and *rework* operations. The work operation is mandatory for each item. The outcome of this operation is the production of either a good quality item or a defective one. Producing a defective item gives rise to the need for the second operation, i.e., the rework operation. After the rework operation, the corresponding item is assumed to be of the required good quality. Both operations are performed on the same machine. Defective items deteriorate while waiting for the rework. This results in increased time and cost for their rework.

Items are manufactured in batches, each preceded by a setup time s_1. Each batch consists of two sub-batches separated by a setup time s_2, see Fig. 1 in Section 3. The first and the second sub-batches include the work and rework operations of the same batch, respectively. The processing times of the work operations are the same. The processing time of a rework operation is a linear increasing function of the holding time of the defective item between its work and rework operations. Without loss of generality, we assume that each work operation requires one unit of time and each rework operation requires $p + a \cdot t$ time, where p and a are non-negative numbers and t is the difference between the starting time of the rework operation and the completion time of the corresponding work operation of the same defective item.

We assume that all the operations in the same sub-batch complete at the same time when the last operation in the sub-batch is finished. This assumption holds in situations where items are manufactured in containers such as boxes, pallets or carts, or where an additional operation such as inspection, sorting, packing or labeling is needed after all the

items in the same sub-batch have been produced.

Flapper *et al.* [1] pointed out that in different process industries input materials may have partly unknown compositions. This leads to uncertainty about the distribution of defective items. However, given adequate historical statistical data, it is possible to reliably estimate and ascertain that defective items on average constitute a stable and relatively fixed percentage of the total number of items produced in any batch. Inderfurth, Lindner and Rahaniotis [7], Teunter and Flapper [11] and Inderfurth *et al.* [6] made this assumption in their studies. We follow their approach by assuming that the percentage of defective items is the same in each batch and it is known. More specifically, we assume that a defective item follows $v-1$ good quality items in every batch. Therefore, if there are x defective items in a batch, then there are $x(v-1)$ good quality items in it, where v is an integer, $v \geq 2$. Note that $\frac{1}{v}$ is, in fact, the percentage of defective items, which is known as *fraction defective* in the quality control literature, see Gitlow *et al.* [2]. Thus, $N = nv$, where n is the total number of defective items to be remanufactured.

Note that our model (as well as the models in [7] and [6]) is a simplification of a real-life situation. In particular, given the fraction defective $\frac{1}{v}$, we allow only batches that contain numbers of items being multiples of v. If this assumption is unacceptable, we can redefine a part of an item as the whole item. For example, redefining $\frac{1}{v}$-th part of an item as the whole item allows any number of (new) items in the batch. In this case, the size of the problem increases and its solution has to be appropriately converted into the solution of the original problem. Existing stochastic models that work with average data rather than with real data, and continuous models that work with real-valued data face the same difficulty. However, more complicated models would demand solution procedures with much higher computational time and space requirements than those presented in the literature and in this paper. Experiments with real data should be performed to determine whether more complicated models are worth being constructed.

We further assume that the machine is an expensive piece of equipment, and therefore, no machine idle time is allowed. This assumption, together with the batch availability assumption, implies that a schedule is completely characterized by a partition of the set of items into batches and their sequence. Furthermore, since all the items are identical and

a defective item follows $v-1$ good quality items, a batch is completely characterized by the number of defective items in it. We shall call the number of defective items in a batch the *size* of the batch. Consequently, a schedule is completely determined by the number of batches and the sequence of batch sizes.

Let us call an item a *finished item* if it is a good quality item after its work operation or if it has become a good quality item after the rework operation. The following costs are considered:

α – batch setup cost,

β – finished item holding cost (earliness cost),

γ – unfinished defective item holding cost.

Given a schedule, consider the corresponding non-decreasing sequence of the completion times of the finished items: $C_1 \leq C_2 \leq \cdots \leq C_N$. We say that a finished item is in position i if its completion time is C_i. Denote by S_i^R and C_i^W the starting time of the rework operation and the completion time of the work operation, respectively, of a defective item in position i.

Let S be a schedule with k batches and D be the set of defective items of the schedule S. The problem is to find an optimal schedule such that the deadlines are satisfied, i.e., $C_i \leq d_i$, $i = 1, \ldots, N$, and the following function of total cost is minimized:

$$F(S) = \alpha k + \beta \sum_{i=1}^{N} (d_i - C_i) + \gamma \sum_{i \in D} (S_i^R - C_i^W).$$

We denote this problem by BWR, meaning Batching-Work-Rework.

The described model in which the items in the same sub-batch complete together is called the *batch availability model* in the literature, see for example, Potts and Kovalyov [8]. Inderfurth *et al.* [6] investigated a problem different from problem BWR, where each item completes immediately when its processing is finished. It is called the *item or job availability model* in the literature, see [8]. This difference between the two models makes the results of Inderfurth *et al.* [6] not applicable to deal with problem BWR.

Stochastic and continuous versions of the problem with item availability have been studied by Teunter and Flapper [11] and Inderfurth, Lindner and Rahaniotis [7], respectively. These problems and problem BWR lie in the area of the optimal planning and control of work and rework processes reviewed by Flapper *et al.* [1], and in the area of the optimal

control of deteriorating production reviewed by Goyal and Giri [4].

In the following section we describe an approach to extending and evaluating partial feasible schedules for the general problem BWR. Then we construct dynamic programming algorithms for problem BWR and its special cases. An $O(n^2 d_N)$ time algorithm for the general problem BWR is described in Section 4. Section 5 presents an $O(\frac{n^B}{(B-1)!})$ time algorithm for the special case where the size of each batch is upper-bounded by B, $1 \leq B \leq n$. Section 6 describes an $O(n^3)$ time algorithm for the special case with zero holding cost for the finished items. In section 7 we present an $O(n^2)$ algorithm for solving the deadline-free case of the problem where all the items are kept till the end of the schedule. The paper concludes with a summary of the results and suggestions for future research.

3 Extending and evaluating partial feasible schedules

Recall that $d_1 \leq \cdots \leq d_N$ and the batch size is the number of defective items in a batch.

Consider a partial feasible schedule S in which there are b defective items. Thus, bv good quality items are produced and the positional deadlines $d_1, \ldots, d_{bv}$ are met. Assume that the last finished item in position bv completes at time t. Let us assign a batch of size j to the end of this schedule and calculate the contribution of the added batch to the objective function. A graphical representation of the described situation is given in Fig. 1.

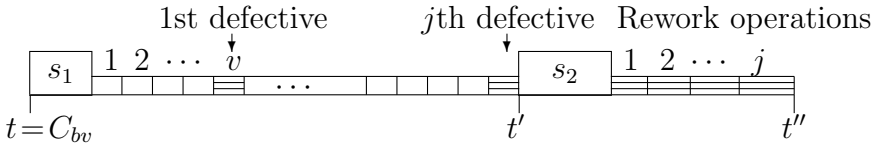

Figure 1: A batch with j defective items.

In this figure, good quality items in positions $bv+1, \ldots, bv+j(v-1)$ complete at time $t' = t + s_1 + jv$ and (reworked) good quality items in positions $bv+j(v-1)+1, \ldots, bv+jv$ complete at time $t'' = t' + s_2 + P(j)$, where $P(j)$ is the processing time of the rework sub-batch.

The contribution of the added batch of size j to the objective function is

$$\Delta(j) = \alpha + \beta\Big(F_q(j) + F_d(j)\Big) + \gamma H(j), \tag{1}$$

where $F_q(j)$, $F_d(j)$ and $H(j)$ are the contribution of good quality items to the total finished items holding time, the contribution of defective items to the total finished items holding time, and the contribution of reworked items to the total unfinished defective item holdings time, respectively.

We have

$$F_q(j) = \sum_{i=1}^{j(v-1)} d_{bv+i} - j(v-1)(t+s_1+jv). \tag{2}$$

In order to find F_d, we first calculate the holding time and the rework operation processing time of each unfinished defective item. Consider a defective item whose rework operation is i-th in the appended batch, $i = 1, \ldots, j$. Let h_i and $p_i = p + ah_i$ be the holding time and the rework operation processing time of this item, respectively. We have

$$h_1 = s_2,\ p_1 = p + ah_1,$$

$$p_i = p + ah_i,\ h_i = h_{i-1} + p_{i-1} = p + (a+1)h_{i-1},\ i = 2, \ldots, j,$$

from which we deduce that

$$h_i = p\frac{(a+1)^{i-1}-1}{a} + (a+1)^{i-1}s_2,\ p_i = (p+as_2)(a+1)^{i-1},\ i = 2, \ldots, j.$$

Then, the processing time of the rework sub-batch is

$$P(j) = \sum_{i=1}^{j} p_i = (p+as_2)\sum_{i=1}^{j}(a+1)^{i-1} = (p+as_2)\frac{(a+1)^j - 1}{a}$$

and the processing time of the appended batch of size j is

$$T(j) = s_1 + jv + s_2 + P(j) = s_1 + jv + s_2 + (p+as_2)\frac{(a+1)^j - 1}{a}. \tag{3}$$

Then

$$F_d(j) = \sum_{i=1}^{j}[d_{bv+j(v-1)+i} - (t+T(j))]$$

and, on substitution of $T(j)$,

$$F_d(j) = \sum_{i=1}^{j} d_{bv+j(v-1)+i} - j(t+s_1+s_2+jv - \frac{p+as_2}{a}) - j\frac{(p+as_2)}{a}(a+1)^j. \tag{4}$$

The contribution to the total unfinished defective item holding time, $H(j)$, is equal to

$$H(j) = \sum_{i=1}^{j} h_i = (a+1)^j\left(\frac{p}{a^2} + \frac{s_2}{a}\right) - \frac{p}{a^2} - \frac{s_2+jp}{a}. \tag{5}$$

In order to determine whether the schedule with the appended batch is feasible, we need to check the following inequalities:

$$t + s_1 + jv \leq d_{bv+1} \tag{6}$$

$$t + T(j) \leq d_{bv+j(v-1)+1}. \tag{7}$$

Inequality (6) ensures that $j(v-1)$ good quality items in the first sub-batch complete by their earliest deadline d_{bv+1}. Similarly, inequality (7) ensures that j reworked items in the second sub-batch complete by their earliest deadline $d_{bv+j(v-1)+1}$.

Our dynamic programming formulation for problem BWR and its special cases constructs schedules by appending a new batch to the end of a partial feasible schedule. Therefore, we shall use equation (1) and its components (2), (4) and (5) in the dynamic programming algorithms to calculate the contribution of an added batch to the objective function.

Let us denote schedule S by a vector $(j_1, j_2, \ldots, j_k)$, in which k is the number of batches and j_i, $1 \leq i \leq k$, is the size of the i-th batch. Denote the cost of schedule S by $F(S)$.

The following lemma can be used to eliminate non-optimal solutions for problem BWR.

Lemma 1 *For any two feasible schedules* $S^{(1)} = (j_1, \ldots, j_{i-2}, j_{i-1}, j_i, j_{i+1}, \ldots, j_k)$ *and* $S^{(2)} = (j_1, \ldots, j_{i-2}, j_i, j_{i-1}, j_{i+1}, \ldots, j_k)$*, in which the* $i-1$*-st and* i*-th batches are interchanged,* $F(S^{(1)}) \leq F(S^{(2)})$ *if and only if* $\frac{T(j_{i-1})}{j_{i-1}} \leq \frac{T(j_i)}{j_i}$.

Proof. It is easy to see that the number of batches and the unfinished defective items holding time are equal in the above two schedules. Thus, the corresponding components of the objective function are also equal. Moreover, the finished items holding time is the same for all the items in $S^{(1)}$ and $S^{(2)}$, except for those that make up the $i-1$-st and i-th batches. Thus, if we denote by $F_d^{(m)}(S)$ and $F_q^{(m)}(S)$ the contribution of the defective and quality items, respectively, of the m-th batch to the finished items holding time in schedule S, then we have

$$F(S^{(l)}) = E + \beta[F_q^{(i-1)}(S^{(l)}) + F_q^{(i)}(S^{(l)}) + F_d^{(i-1)}(S^{(l)}) + F_d^{(i)}(S^{(l)})], l = 1, 2,$$

where E is the total cost of schedule $S^{(l)}$ without the contribution of the items from the $i-1$-st and i-th batches to the finished items holding cost.

Assume that partial schedule $(j_1, j_2, \ldots, j_{i-2})$ contains b defective items and its completion time is t. Using equations (2) and (4), we have

$$\begin{aligned} F(S^{(1)}) &= E + \beta[F_q^{(i-1)}(S^{(1)}) + F_q^{(i)}(S^{(1)}) + F_d^{(i-1)}(S^{(1)}) + F_d^{(i)}(S^{(1)})] = \\ & E + \beta[\sum_{r=1}^{j_{i-1}(v-1)} d_{bv+r} - j_{i-1}(v-1)(t+s_1+j_{i-1}v) + \sum_{r=1}^{j_i(v-1)} d_{(b+j_{i-1})v+r} - \\ & j_i(v-1)(t+T(j_{i-1})+s_1+j_iv) + \sum_{r=1}^{j_{i-1}} d_{bv+j_{i-1}(v-1)+r} - j_{i-1}(t+T(j_{i-1})) + \\ & \sum_{r=1}^{j_i} d_{(b+j_{i-1})v+j_i(v-1)+r} - j_i(t+T(j_{i-1})+T(j_i))] = E + \beta[A - vj_iT(j_{i-1})], \end{aligned}$$

where

$$\begin{aligned} A &= \sum_{r=1}^{(j_{i-1}+j_i)v} d_{bv+r} - (t+s_1)(v-1)(j_{i-1}+j_i) - (v-1)v(j_{i-1}^{(2)} + j_i^{(2)}) - t(j_{i-1}+j_i) - \\ & j_{i-1}T(j_{i-1}) - j_iT(j_i). \end{aligned}$$

Similarly,

$$F(S^{(2)}) = E + \beta[F_q^{(i-1)}(S^{(2)}) + F_q^{(i)}(S^{(2)}) + F_d^{(i-1)}(S^{(2)}) + F_d^{(i)}(S^{(2)})] = E + \beta[A - vj_{i-1}T(j_i)].$$

Now, from the expressions for $F(S^{(1)})$ and $F(S^{(2)})$, we find that $F(S^{(2)}) - F(S^{(1)}) = \beta v(j_{i-1}T(j_i) - j_iT(j_{i-1}))$. Inequality $F(S^{(1)}) \leq F(S^{(2)})$ is equivalent to $\frac{T(j_{i-1})}{j_{i-1}} \leq \frac{T(j_i)}{j_i}$, as required. ∎

Let $S = (j_1, \ldots, j_k)$ be an optimal schedule. It follows from Lemma 1 that for any i, $2 \leq i \leq k$, either $\frac{T(j_{i-1})}{j_{i-1}} \leq \frac{T(j_i)}{j_i}$ or schedule $(j_1, \ldots, j_{i-2}, j_i, j_{i-1}, j_{i+1}, \ldots, j_k)$ is not feasible. This fact can be used to curtail the number of prospective extensions of a partial schedule. Let $S^{(1)} = (j_1^{(1)}, \ldots, j_k^{(1)})$ be a partial schedule. Schedule $(S^{(1)}, j)$ cannot be extended to an optimal one if $\frac{T(j_k^{(1)})}{j_k^{(1)}} > \frac{T(j)}{j}$ and schedule $(j_1^{(1)}, \ldots, j_{k-1}^{(1)}, j, j_k^{(1)})$ is feasible. This elimination criterion can be incorporated into the dynamic programming algorithms to curtail the number of prospective extensions of a partial schedule.

4 Dynamic programming algorithm for problem BWR

The computational complexity of problem BWR is unknown. We present a pseudopolynomial dynamic programming algorithm to solve the problem. The algorithm uses the

completion time of a partial schedule and the number of defective items in it as state variables.

Let $Z(t,b)$, $0 \leq t \leq d_N$, $0 \leq b \leq n$, denote the class of partial feasible schedules completing at time t and containing b defective items. The proposed dynamic programming algorithm enumerates values t and b, $0 \leq t \leq d_N$, $0 \leq b \leq n$. Let $F(t,b)$ denote the cost of a schedule in class $Z(t,b)$. A formal description of the algorithm is given below.

Algorithm for problem BWR

Step 1 (Initialization) Set $F(0,0) = 0$. Set $F(t,b) = \infty$ for $t = 0, 1, \ldots, d_N$, $b = 0, 1, \ldots, n$, and $(t,b) \neq (0,0)$. Calculate $T(j)$ and $\Delta(j)$, $j = 1, \ldots, n$, according to (3) and (1).

Step 2 (Recursion) For $t = 1, \ldots, d_N$ and $b = 1, \ldots, n$, compute the following

$$F(t,b) = \min_{j=1,\ldots,b} \begin{cases} F(t - T(j), b - j) + \Delta(j), & \text{if (6) and (7) are satisfied,} \\ \infty, & \text{otherwise.} \end{cases}$$

Step 3 (Optimal solution) Calculate the optimal solution value

$$F^* = \min\{F(t,n) \mid 1 \leq t \leq d_N\}.$$

If $F^* = \infty$, then there is no schedule feasible with respect to the deadlines $d_1, \ldots, d_N$.

If $F^* < \infty$, then the corresponding optimal schedule can be found by backtracking.

To evaluate the time complexity of the above algorithm, we note that the number of state variables is $O(nd_N)$ and the algorithm determines each value $F(t,b)$ in $O(n)$ time. Thus, the time complexity of the proposed algorithm is $O(n^2 d_N)$ and, consequently, problem BWR is not NP-hard in strong sense.

Note that we can assume without loss of generality that values $T(1), \ldots, T(n)$ are relatively prime. Otherwise, the time complexity of the above algorithm can be reduced to $O(n^2 \frac{d_N}{GCD(T(1),T(2),\ldots,T(n))})$, where $GCD(m_1, m_2, \ldots, m_n)$ is the greatest common divisor of the numbers $m_1, m_2, \ldots, m_n$.

5 The case of limited batch size

In this section we assume that each batch size (i.e., number of defective items in each batch) is upper-bounded by B, $1 \leq B \leq n$. In many applications, the upper bound B is associated

with a physical limit on the capacity of the containers that carry the manufactured items. We denote this problem by Lim-BWR, meaning Limited-Batching-Work-Rework.

Denote by $Z(l_1,\ldots,l_B)$ the class of partial schedules, where each schedule contains l_i batches of size i, $i=1,\ldots,B$. Since the length of a batch is determined by the number of defective items in it, all the schedules from $Z(l_1,\ldots,l_B)$ have the same length. Denote this length by $C(l_1,\ldots,l_B)$, which is:

$$C(l_1,\ldots,l_B)=\sum_{j=1}^{B}T(j)l_j, \tag{8}$$

where $T(j)$ is calculated according to (3).

Note that all the schedules in class $Z(l_1,\ldots,l_B)$ have the same number of defective items. Denote this number by $b=\sum_{j=1}^{B}l_jj$.

Let $S\in Z(l_1,\ldots,l_B)$ be a partial feasible schedule. Consider schedule $S^{(1)}=(S,j)$, in which j, $1\le j\le\min\{B,n-b\}$, is the size of the appended batch. Schedule $S^{(1)}$ is feasible if and only if inequalities (6) and (7) are satisfied, where $t=C(l_1,\ldots,l_B)$.

We have $S^{(1)}\in Z(l_1,\ldots,l_j+1,\ldots,l_B)$. Calculate the cost $F(S^{(1)})=F(S)+\Delta(j)$. Value $\Delta(j)$ and its components F_q, F_d and H are calculated according to (1), (2), (4) and (5), respectively, where $C=C(l_1,\ldots,l_B)$.

The proposed dynamic programming algorithm enumerates vectors $(l_1,\ldots,l_B)$. Value $F(l_1,\ldots,l_B)$ represents the minimum cost of a partial schedule in class $Z(l_1,\ldots,l_B)$. The description of the algorithm is as follows.

Algorithm for problem Lim-BWR

Step 1 (Initialization) F(0,. . .,0)=0. Set $F(l_1,\ldots,l_B)=\infty$ for all $(l_1,\ldots,l_B)$ such that $0<\sum_{j=1}^{B}l_jj\le n$. Calculate $T(j),j=1,\ldots,B$, using (3) and calculate $\Delta(j),j=1,\ldots,B$, using (1). Calculate $C(l_1,l_2,\ldots,l_B)$ for all $(l_1,l_2,\ldots,l_B)$ such that $0\le\sum_{j=1}^{B}l_jj\le n$ using (8).

Step 2 (Recursion) For $b=1,\ldots,n$ and $(l_1,\ldots,l_B)$ such that $\sum_{j=1}^{B}l_jj=b$, compute

$$F(l_1,\ldots,l_B)=\min_{j=1,\ldots,\min\{b,B\}}\begin{cases}F(l_1,\ldots,l_j-1,\ldots,l_B)+\Delta(j), & \text{if } l_j>0,\ (6)\text{ and }(7)\text{ are}\\ & \text{satisfied for } t=C(l_1,\ldots,l_B),\\ \infty, & \text{otherwise.}\end{cases}$$

Step 3 (Optimal solution) Calculate the optimal solution value

$$F^* = \min\{F(l_1, \ldots, l_B) \mid \sum_{j=1}^{B} l_j j = n\}.$$

If $F^* = \infty$, then there is no schedule feasible with respect to the deadlines $d_1, \ldots, d_N$.

If $F^* < \infty$, then the corresponding optimal schedule can be found by backtracking.

Since $l_i \leq n/i, i = 1, \ldots, B$, the number of state variables used by the proposed algorithm is $O(\frac{n^B}{B!})$. We need $O(B)$ time to calculate $C(l_1, l_2, \ldots, l_B)$ for each state and therefore, the preprocessing in Step 1 takes $O(\frac{n^B}{(B-1)!})$ time. Furthermore, the algorithm considers $O(B)$ states to determine each $F(l_1, \ldots, l_B)$. Thus, Step 2 runs in $O(\frac{n^B}{(B-1)!})$ time. Finally, Step 3 takes $O(\frac{n^B}{B!})$ time. Consequently, the total time complexity of the algorithm is $O(\frac{n^B}{(B-1)!})$, which is polynomial if B is a constant.

6 The case of zero holding costs for the finished items

Assume that all the finished items are held at zero cost, i.e., $\beta = 0$. Denote this problem by ZeroHold-BWR.

Let $C(S)$ and $F(S)$ be the length and cost of schedule S, respectively. The following lemma establishes a relation between two schedules with the same number of batches and the same number of defective items.

Lemma 2 *Let $S^{(1)} = (j_1^{(1)}, \ldots, j_k^{(1)})$ and $S^{(2)} = (j_1^{(2)}, \ldots, j_k^{(2)})$ be two schedules with k batches and b defective items each. We have $C(S^{(1)}) \leq C(S^{(2)})$ if and only if $F(S^{(1)}) \leq F(S^{(2)})$.*

Proof. First, consider a schedule $S = (j_1, j_2, \ldots, j_k)$. An analysis of formulas (5) and (1) shows that in the case where $\beta = 0$ the contribution of a batch added to the end of a partial schedule solely depends on its size. Therefore, we have $F(S) = \sum_{i=1}^{k} \Delta(j_i)$, where $\Delta(j_i) = \alpha + \gamma H(j_i)$ and $H(j_i)$ is given in (5), $i = 1, \ldots, k$. Calculate

$$\begin{aligned} F(S) &= \sum_{i=1}^{k} \Delta(j_i) = \alpha k + \gamma \sum_{i=1}^{k} [(a+1)^{j_i} \left(\frac{p}{a^2} + \frac{s_2}{a}\right) - \frac{p}{a^2} - \frac{s_2 + j_i p}{a}] = \\ & A_1^{(1)} k + A_2^{(1)} \sum_{i=1}^{k} (a+1)^{j_i} + A_3^{(1)} \sum_{i=1}^{k} j_i, \end{aligned}$$

where

$$A_1^{(1)} = \alpha - \gamma\left(\frac{p}{a^2} + \frac{s_2}{a}\right), A_2^{(1)} = \frac{p}{a^2} + \frac{s_2}{a}, A_3^{(1)} = -\frac{p}{a}.$$

Next, calculate $C(S) = \sum_{i=1}^{k} T(j_i)$. On substituting $T(j_i), i = 1, \ldots, k$, obtained from (3), into $C(S)$, we deduce that

$$\begin{aligned} C(S) &= \sum_{i=1}^{k} T(j_i) = \sum_{i=1}^{k}\left(s_1 + j_i v + s_2 + (p + as_2)\frac{(a+1)^{j_i} - 1}{a}\right) = \\ & A_1^{(2)}k + A_2^{(2)}\sum_{i=1}^{k}(a+1)^{j_i} + A_3^{(2)}\sum_{i=1}^{k} j_i, \end{aligned}$$

where

$$A_1^{(2)} = s_1 + s_2 - \frac{p + as_2}{a}, A_2^{(2)} = \frac{p + as_2}{a}, A_3^{(2)} = v.$$

To complete the proof, we use the above expressions of $F(S)$ and $C(S)$ in the following chain of equivalent inequalities.

$$C(S^{(1)}) \le C(S^{(2)}) \Leftrightarrow A_1^{(2)}k + A_2^{(2)}\sum_{i=1}^{k}(a+1)^{j_i^{(1)}} + A_3^{(2)}b \le A_1^{(2)}k + A_2^{(2)}\sum_{i=1}^{k}(a+1)^{j_i^{(2)}} + A_3^{(2)}b \Leftrightarrow$$

$$\Leftrightarrow \sum_{i=1}^{k}(a+1)^{j_i^{(1)}} \le \sum_{i=1}^{k}(a+1)^{j_i^{(2)}} \Leftrightarrow$$

$$\Leftrightarrow A_1^{(1)}k + A_2^{(1)}\sum_{i=1}^{k}(a+1)^{j_i^{(1)}} + A_3^{(1)}b \le A_1^{(1)}k + A_2^{(1)}\sum_{i=1}^{k}(a+1)^{j_i^{(2)}} + A_3^{(1)}b \Leftrightarrow F(S^{(1)}) \le F(S^{(2)}).$$

■

Lemma 2 forms the basis of a polynomial dynamic programming algorithm to solve problem ZeroHold-BWR. Let $Z(k, b)$, $0 \le k \le n$, $0 \le b \le n$ denote the class of partial schedules, where each schedule contains k batches and b defective items. Consider $S \in Z(k, b)$. Assume that S is feasible. Schedule $S^{(1)} = (S, j)$ is feasible if and only if inequalities (6) and (7) are satisfied. Its length can be calculated as $C(S^{(1)}) = C(S) + T(j)$, where $T(j)$ is given in (3).

The proposed dynamic programming algorithm enumerates vectors (k, b). Value $C(k, b)$ represents the minimum length of a partial schedule in the class $Z(k, b)$.

Algorithm for problem ZeroHold-BWR

Step 1 (Initialization) Set $C(0, 0) = 0$ and $C(k, b) = \infty$ for $k = 0, \ldots, n$, $b = 0, \ldots, n$, $(k, b) \ne (0, 0)$. Calculate $T(j), j = 1, \ldots, n$, using (3).

Step 2 (Recursion) Given (k, b), $k = 1, \ldots, n$, $b = k, \ldots, n$, compute $t = C(k-1, b-j)$ and

$$C(k,b) = \min_{j=1,\ldots,b} \begin{cases} t + T(j), & \text{if (6) and (7) are satisfied,} \\ \infty, & \text{otherwise.} \end{cases}$$

Step 3 (Optimal solution) Calculate the minimum length

$$C^* = \min\{C(k,n) \mid 1 \le k \le n\}.$$

If $C^* = \infty$, then there is no schedule feasible with respect to the deadlines $d_1, \ldots, d_N$. If $C^* < \infty$, then the corresponding optimal schedule can be found by backtracking. Let it be $S^* = (j_1^*, \ldots, j_k^*)$. Due to Lemma 2, $F^* = \sum_{i=1}^{k} \Delta(j_i^*)$ is the minimum cost for problem ZeroHold-BWR.

The number of state variables used by this algorithm is $O(n^2)$. The initialization in Step 1 takes $O(n^2)$ time and the backtracking in Step 3 takes $O(n^2)$ time. Since the algorithm demands $O(n)$ time to calculate $C(k, b)$ in Step 2, its time complexity is $O(n^3)$.

7 The deadline-free case

Consider another case of problem BWR. There are no deadlines; however, all the processed items are kept till the required quantities of good quality items are produced. This situation arises, for example, when a manufacturer performs long-term planning and wants to determine the amount of time needed to fulfill the order of a single customer, and to determine the structure of such an optimal schedule. This scheme is also applicable when a manufacturer runs through a low demand period of the year and receives orders from customers one by one. In this case, the time intervals between adjacent orders allow the manufacturer to adjust the production schedule for each particular customer. Denote this problem by Deadline-Free-BWR.

Denote by $Z(b)$ the class of partial schedules, each of which contains b defective items. Assume that S is the schedule from the class $Z(b)$. Let $F(S)$ be the cost of S. Consider schedule $S^{(1)} = (S, j)$ and find $F(S^{(1)}) = F(S) + \Delta(j)$, where $\Delta(j)$ can be calculated as

$$\Delta(j) = \alpha + \beta\Big(T(j)bv + (T(j) - s_1 - jv)(j-1)v\Big) + \gamma H(j), \tag{9}$$

where $T(j)$ and $H(j)$ are found using (3) and (5), respectively. The first component α of the sum (9) is the setup cost of the appended batch. The second component represents the increase in the finished items holding cost. First, this increase is caused by the prolonged holding of the items assigned to schedule S: with a new batch having been added, all bv items of schedule S are to be kept for an additional time $T(j)$. Second, $(j-1)v$ good quality items of the added batch are held for $T(j) - s_1 - jv$ time units until schedule $S^{(1)}$ has been completed. Finally, the third component $\gamma H(j)$ in (9) is the holding cost of the unfinished defective items assigned to be processed in the added batch.

The proposed dynamic programming algorithm enumerates values $b, 0 \leq b \leq n$. Let $F(b)$ be the cost of a partial schedule in class $Z(b)$. The description of the algorithm is given below.

Algorithm for the problem Deadline-Free-BWR

Step 1 (Initialization) Set $F(0) = 0$. Set $F(b) = \infty, b = 1, \ldots, n$. Calculate $\Delta(j)$, $j = 1, \ldots, n$, using (9).

Step 2 (Recursion) For $b = 1, \ldots, n$, compute

$$F(b) = \min_{j=1,\ldots,b} \{F(b-j) + \Delta(j)\}.$$

Step 3 (Optimal solution) The optimal solution cost is equal to $F(n)$. The corresponding schedule can be found by backtracking. ∎

The initialization in Step 1, as well as the finding of an optimal solution in Step 3, takes $O(n)$ time. Furthermore, the proposed algorithm demands $O(n)$ time to determine each $F(b), b = 1, \ldots, n$. Therefore, the complexity of the algorithm is $O(n^2)$. Note that it is pseudopolynomial for problem Deadline-Free-BWR because the length of its input is $O(\log \max\{s_1, s_2, \alpha, \beta, \gamma, v, n\})$.

8 Conclusions

The problem of scheduling the production of new and recoverable defective items of the same product manufactured on the same facility was studied. The situation was modelled

as a problem of scheduling N items, subject to satisfying a set of given order deadlines. We presented an algorithm polynomial in n (the number of defective items) and d_N (the time of the latest demand for a good quality item) for solving the general problem. We also constructed algorithms polynomial in n for solving three realistic special cases of the general problem.

Investigation of other efficiently solvable and practically relevant special cases of the problem is of interest for future research. An open question is the computational complexity of the general problem and its deadline-free case. Further research addressing these issues is worth undertaking.

9 Acknowledgments

We are grateful to three anonymous referees for their constructive comments on earlier versions of our paper. This research was supported in part by The Hong Kong Polytechnic University under a grant from the *Area of Strategic Development in China Business Services*. In addition, M.S. Barketau and M.Y. Kovalyov were supported by INTAS under grant number 03-51-5501.

References

[1] Flapper S.D.P., Fransoo J.C., Broekmeulen R.A.C.M. and Inderfurth K. (2002) Planning and control of rework in the process industries: a review, *Production Planning & Control* **1** 26-34.

[2] Gitlow H., Gitlow S., Oppenheim A. and Oppenheim R. (1989) *Tools and Methods for the Improvement of Quality*, Irwin, Boston.

[3] Gordon V.S. and Kubiak W. (1998) Single machine scheduling with release and due date assignment to minimize the weighted number of late jobs, *Information Processing Letters* **68** 153-159.

[4] Goyal S.K. and Giri B.C. (2001) Recent trends in modeling of deteriorating inventory, *European Journal of Operational Research* **134** 1-16.

[5] Hall N.G., Sethi S.P. and Sriskandarajah C. (1991) On the complexity of generalized due date scheduling problems., *European Journal of Operational Research* **51** 100-109.

[6] Inderfurth K., Kovalyov M.Y., Ng C.T. and Werner F. (2006) Cost minimizing scheduling of work and rework processes on a single facility under deterioration of reworkables, *International Journal of Production Economics*, to appear.

[7] Inderfurth K., Lindner G. and Rahaniotis N.P. (2003) Lotsizing in a production system with rework and product deterioration. Preprint 1/2003. Faculty of Economics and Management, Otto-von-Guericke-University Magdeburg, Germany.

[8] Potts C.N. and Kovalyov M.Y. (2000) Scheduling with batching: a review, *European Journal of Operations Research* **120** 228-249.

[9] Qi X.T., Yu G. and Bard J.F. (2002) Single machine scheduling with assignable due dates, *Discrete Applied Mathematics* **122** 211-233.

[10] Tanaka K. and Vlach M. (1999) Minimizing maximum absolute lateness and range of lateness under generalized due dates on a single machine, *Annals of Operations Research* **86** 507-526.

[11] Teunter R.H. and Flapper S.D.P. (2003) Lot-sizing for a single-stage single-product production system with rework of perishable production defectives, *OR Spectrum* **25** 85-96.