INSEEC Research

PARIS • BORDEAUX • LYON • ALPES-SAVOIE
MONACO • LONDON • CHICAGO

Working Paper
n°: 2014-100-01

Multi-Product Batching and Scheduling with Buffered Rework: The Case of a Car Paint Shop

Florian Jaehn [a], Sergey Kovalev [b], Mikhail Y. Kovalyov [c], Erwin Pesch [d]

a Institute of Sustainable Operations and Logistics at the University of Augsburg, Germany
b INSEEC Business School ECE Lyon, 19 place Tolozan, 69001 Lyon, France
c United Institute of Informatics Problems, National Academy of Sciences of Belarus, Surganova 6, 220012 Minsk, Belarus
d Institute of Information Systems at the University of Siegen, Germany

February 2014

An ulterior version of this article appeared in Naval Research Logistics, vol 61, Issue 6, September 2014, ISSN 1520-6750
It can be purchased at: http://onlinelibrary.wiley.com/doi/10.1002/nav.21596/pdf

Abstract

We study a problem of scheduling products on the same facility, which is motivated by a car paint shop. Items of the same product are identical. Operations on the items are performed sequentially in batches, where each batch is a set of operations on the same product. Some of the produced items are of the required good quality and some items can be defective. Defectiveness of an item is determined by a given simulated function of its product, its preceding product, and the position of its operation in the batch. Defective items are kept in a buffer of a limited capacity, and they are then remanufactured at the same facility. A minimum waiting time exists for any defective item before its remanufacturing can commence. Each product has a sequence independent setup time which precedes its first operation or its operation following an operation of another product. A due date is given for each product such that all items of the same product have the same due date and the objective is to find a schedule which minimizes maximum lateness of product completion times with respect to their due dates. The problem is proved NP-hard in the strong sense, and a heuristic Group Technology solution approach is suggested and analyzed. The results justify application of the Group Technology approach to scheduling real car paint shops with buffered rework.

Key Words: production, scheduling, batching, rework, group technology, car painting.

1 Motivation and literature review

Consider a painting line for the car bodies of various given models and colors. We call a pair (car model, color) a *product* and a car body of a given car model and color an *item* of the corresponding product. Items are painted sequentially and setups are required between products to clean the painting equipment and working area and prepare for the next product. The setup time depends only on the next product because cleaning takes the same time and the preparation steps are product dependent.

There is a quality inspection after which an item either goes to the inventory of good quality items or to the buffer of defective items, which has a limited capacity. The defects can be spots of wrong colors or shades, non-smooth surface, and absence or low thickness of paint. They can be due to the insufficient cleaning of the spray guns, catching dust pieces by the paint, the imperfect positioning of spray heads or incorrect painting time.

If the buffer is not empty, then, after a certain number of items has been produced, it can be decided that some or all defective items leave the buffer, enter the line and are repainted. Any item can be repainted several times. Thus, the production of any item includes one *work operation* and it may include several *rework operations*. Though a rework operation can skip some sub-operations of the paint shop, work and rework operations on the same product require the same time, which is the conveyor pace time for this product. If an operation results in a defective item, we call such operation *defective*.

A maximal set of operations on items of the same product worked and reworked since last setup is called a *batch*. *Batch size* is the number of operations in this batch. Usually, operations in the beginning of a batch result in a defective item more often than those in the end of the batch, because the negative impact of the previous color and the setup operations is smaller in the end of the batch.

It is assumed that defectiveness of an item is a function of its product, its preceding product on the line, and position of its operation in the batch. Pawlak and Rucinski [31] suggested that these characteristics are main factors affecting the item's quality. We suggest that the defectiveness functions is simulated based on the historical data.

It is also assumed that there is a minimum time for any defective item to stay in the buffer, which is needed for the paint to dry.

Each good quality item has a *due date* which depends on the time of the delivery of the corresponding car to a client. We assume that all items of the same product have the same due date. This assumption may require creating several groups for car bodies of the same model and color. The objective is to construct a work/rework schedule such that the maximum lateness of items with respect to their due dates is minimized.

The due date satisfaction criterion is important in a make-to-order environment, where item delivery times are negotiated with the clients. While the just-in-time delivery is the most desirable for the clients, it is often in conflict with the cost of setting the production up. In many cases the clients accept receiving their orders before the due dates. In this case, minimizing the maximum lateness is an appropriate criterion. In many situations, the cost of non-production breaks, including the setups, considerably overweight the inventory holding costs. Since minimizing the maximum lateness indirectly minimizes the non-production breaks, it also minimizes the overall cost in these situations.

A schedule will be characterized by a sequence of batches. Each batch will be characterized by its product and the number of non-defective and defective operations. In a batch, positions of non-defective and defective operations are uniquely determined by the defectiveness function. A given operation can be arbitrarily specified as a work operation on a new item or as a rework operation on an item from the buffer if it spent sufficient time there.

Pawlak et al. [29] and Pawlak and Rucinski [31] observed the described situation in a real car paint shop. They discussed factors that influence appearance of defects and suggested on-line solution procedures for minimizing the makespan and the number of color changes. In this paper we assume that all data are given. Solution approaches to this deterministic problem, which is already computationally hard, can be used to solve a more realistic problem with stochastic or uncertain parameters.

As part of a more general car sequencing problem observed in *Groupe Renault*, a car paint shop scheduling problem was studied by Gagné et al. [11], Cordeau et al. [7], Joly and Frein [22], Ribeiro et al. [34], Solnon et al. [38], Morin et al. [28], Giard and Jeunet [12], Golle et al. [15] and Zinflou et al. [44]. The due date satisfaction criterion for a car sequencing problem was studied by Guerre-Chaley et al. [16]. Boysen et al. [4] noticed that the due dates are important in an assembly-to-order environment. The car production process considered in these studies includes three main stages: body production, body painting and car assembling in this order. Solnon et al. [38] assumed that the same color bodies are produced consecutively and that the production sequence is the same for the paint shop and the assembly shop. Guerre-Chaley et al. [16], Spieckermann et al. [39] and Pawlak et al. [30] considered buffers between the stages, which are used for re-sequencing car bodies. Meyr [26] wrote that buffers between stages are necessary because failures in the body and paint shops occur frequently. According to Holweg [18] cited by Meyr [26], the rework rate can be up to 40-50%. Computational complexity of a problem, in which a sequence of car bodies of different models is given and a decision has to be made about the assignment of colors to the car bodies, was studied by Epping et al. [9], Bonsma et al. [3], and Meunier and Sebó [25]. To the best of our knowledge, car scheduling models with buffered rework have been studied exclusively by Pawlak et al. [29] and Pawlak and Rucinski [31].

The problem under study can be classified into four categories: single machine batch scheduling, scheduling with intermediate buffers, lot-sizing for imperfect production systems, and reverse logistics. Recent surveys of results on batch scheduling are given by Potts and Kovalyov [32] and Allahverdi et al. [2]. Scheduling models with intermediate buffers were analyzed, among others, by Sawik [37], Agnetis et al. [1], and Wang and Tang [42]. Lot-sizing models for the imperfect production systems were considered by Rosenblatt and Lee [35], Liu and Yang [24], Inderfurth et al. [19], Buscher and Lindner [5], Inderfurth et al. [20], to name a few. Within the framework of reverse logistics, problems of batching work and rework processes are surveyed by Flapper et al. [10], Inderfurth and Teunter [21] and de Brito and Dekker [8]. Most of the research in the latter two fields was concentrated on a production environment, in which work and rework processes are performed on the same facility.

Main characteristics of the problem in this paper are that the considered production is essentially discrete, defective items are stored in a buffer of a limited capacity, a lower bound on the storage time is given, there are several products, product dependent setup times are given, and no deterioration occurs to the defective items. To the best of our knowledge, this combination has never been studied in literature before.

In the following section we give required notations and definitions, and discuss the suggested model. Proofs of strong NP-hardness of two important special cases of the problem are presented in Section 3. A heuristic *Group Technology (GT)* approach to solving the problem is described and analyzed in Section 4. We establish properties which are sufficient for the optimality of a GT solution. Most of these properties are satisfied in real car paint shops. Furthermore, a GT approach delivers good quality solutions, which is demonstrated by computational experiments in Section 5. Section 6 contains final remarks, discussion of a stochastic case and suggestions for future research.

2 Notations, definitions and discussion of the model

The following notations will be used.

F - number of products;

n_f - demand (number of good quality items) of product f;

p_f - processing time requirement for any item of product f, the same value for work and rework;

s_f - the setup time required to start a batch of operations on product f if it is sequenced first on the line or immediately after another product;

d_f - the due date for the demand satisfaction of product f;

C_f - the completion time of the last good quality item of product f in a given schedule;

$L_f = C_f - d_f$ - the lateness of product f in a given schedule;

$L_{\max} = \max\{L_f \mid f = 1, \ldots, F\}$ - the maximum lateness of products in a given schedule (objective function to be minimized);

B - the capacity of the buffer;

M - the minimum time that any defective item should stay in the buffer (the *buffer lower time limit*);

$G(g, f, r)$ - a 0-1 function of the product index g of the preceding batch, the product index f of the current batch, and the position r of an operation in this batch such that $G(g, f, r) = 1$ if this operation is defective, and $G(g, f, r) = 0$, otherwise. Here $g \in \{0, 1, \ldots, F\}$, $f \in \{1, \ldots, F\}$, and $g = 0$ applies for the case in which f is the first product on the line. Notice that, given g and f, function G values are the same for the same positions of different batches;

V_f, U_f - the lower and upper bounds, respectively, on the batch size for product f. Thus, $r \in \{1, 2, \ldots, U_f\}$ for the function $G(g, f, r)$. We say that the batch sizes are *unbounded* if $V_f = 1$ and $U_f = \infty$, $f = 1, \ldots, F$.

The upper bounds U_f can be used to model the requirement that the painting equipment should be cleaned after a certain number of painting operations, see Gagné et al. [11].

Assuming that the values of $G(g, f, r)$ are known in advance is a simplification. In practical applications, function G can be simulated by employing a historical data analysis and a

Monte-Carlo method. Firstly, historical data can be used to forecast the probability that $G(g, f, r) = 1$ for all relevant g, f and r. Then this probability can be used in a Monte-Carlo method to simulate the $G(g, f, r)$ value, for all relevant g, f and r. If the quality of real operations seriously deviates from the simulated data, then the function $G(g, f, r)$ can be corrected on-line, and the problem can be re-solved according to the scheduled positions of the defective items.

Most of the existing stochastic work-rework models consider only the fraction of defective operations in each batch and do not characterize the structure of the sequence of operations. They assume that there are few, often one or two, stochastic parameters, which are related to the equipment breakdown time, repair time or preventive maintenance time, see, for example, Chiu [6] and Wee and Widayadana [43]. Teunter and Flapper [40] consider a single-product system with rework and assume that there is the same probability for each item to be defective. The main reason for these simplifying assumptions is that an adequate stochastic problem is too complicated.

We assume that the time of transporting an item to or from the buffer is equal to zero. Furthermore, any item that has stayed in the buffer for at least M time units can leave the buffer (there is a direct access to any item in the buffer). No two items can be manufactured or remanufactured at the same time, and no item can be manufactured or remanufactured while setting the line up. Since the line is an expensive equipment, no idle time is allowed if there is an item to be processed or a setup to be performed at this time. However, the line can be blocked by a defective item if: 1) the buffer is full, and 2) no item can leave the buffer because no item has stayed there for at least M time units. If the line is blocked, no item can be manufactured or remanufactured. However, a setup can be performed even if the line is blocked. A situation when no setup is performed, the buffer is full, a defective item blocks the line and there is an item to be manufactured but it cannot because the line is blocked is called a *deadlock*. This situation should be avoided. We do not consider a situation that could occur at the end of production, in which there is no item to be manufactured and the buffer is not empty, to be a deadlock. The deadlock can *always* be avoided if the total setup and production time between the completion of a defective item and the completion of B-th defective item following this item is at least M for *any* feasible sequence of manufacturing/remanufacturing operations. For example, the deadlock can always be avoided if the batch sizes are unbounded and $M \leq B \cdot p_{\min}$, where $p_{\min} = \min\{p_f \mid f = 1, \ldots, F\}$. The problem is to construct a feasible manufacturing/remanufacturing schedule such that no deadlock occurs (if such a schedule exists), all the demands are satisfied with no overproduction, and the maximum lateness $L_{\max}$ is minimized. We denote this problem as $P(L_{\max})$.

Deadlock handling methods in computer and manufacturing systems were discussed and a deadlock avoidance scheme was presented by Valckenaers and Van Brussel [41], who used in their developments data from an existing car paint shop in Sindelfingen (Germany). Note that an ideal implementation of a feasible solution to the problem $P(L_{\max})$ will not need a deadlock handling mechanism.

Due to the fact that items of the same product are identical, it can easily be seen that a search for an optimal solution can be limited to schedules in which defective items of the same product leave the buffer in the same order as they enter it, following the well known *First-In-First-Out (FIFO)* strategy. A managerial implication of this observation is that the buffer can be designed as a collection of unidirectional lines each of which is dedicated to a specific product.

An example of a schedule for the problem $P(L_{\max})$ is given in Fig. 1. In the corresponding problem, $F = 2$, $n_1 = 10$, $n_2 = 9$ and $B = 2$. It is assumed that $M \leq B \cdot p_{\min} = 2\min\{t_3 - t_2, t_8 - t_7\}$. Therefore, no deadlock occurs.

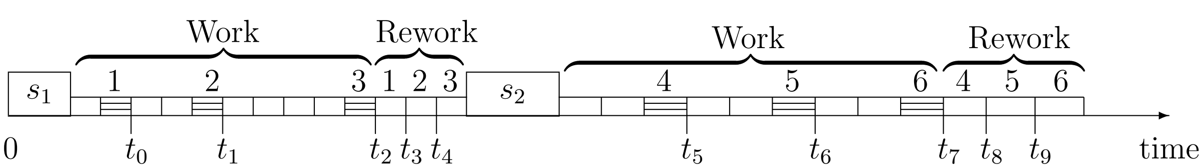

Figure 1: An example of a schedule. Hatched boxes represent defective operations.

Let vector $b(t) = (b_1(t), \ldots, b_F(t))$ describe the buffer content at time t, where $b_f(t)$ is the number of items of product f in the buffer at time t. For the example in Fig. 1, $b(t) = (0, 0)$ for $t < t_0$ and $t \geq t_9$. Other values of $b(t)$ are given in Table 1.

We denote the problem of deciding whether there exists a feasible schedule for the problem $P(L_{\max})$ as the problem *Decide-Deadlock*. We refer to the problem *No-Deadlock* as the

$t \in$	$[t_0, t_1)$	$[t_1, t_2)$	$[t_2, t_3)$	$[t_3, t_4)$	$[t_4, t_5)$	$[t_5, t_6)$	$[t_6, t_7)$	$[t_7, t_8)$	$[t_8, t_9)$
$b(t)$	(1,0)	(2,0)	(2,0)	(1,0)	(0,0)	(0,1)	(0,2)	(0,2)	(0,1)

Table 1: Values of $b(t) = (b_1(t), b_2(t))$ for the example in Fig. 1.

special case of the problem $P(L_{\max})$ in which there exists a feasible schedule. In the following section we will prove that each of the problems *Decide-Deadlock* and *No-Deadlock* is NP-hard in the strong sense. Therefore, the general problem $P(L_{\max})$ is NP-hard in the strong sense as well.

If the production is perfect, then no buffer is needed and the problem $P(L_{\max})$ becomes a classic family scheduling problem with identical jobs in each family, see Potts and Van Wassenhove [33] and Potts and Kovalyov [32]. For this problem, Santos [36] has proved that there exists an optimal solution in which a single batch is made for each product and the products are sequenced in the non-decreasing order of their due dates.

Note that if there is no deadlock for a schedule constructed for a simulated defectiveness function, it can occur for the real defectiveness function.

3 Proving strong NP-hardness

Theorem 1 *The problem Decide-Deadlock is NP-hard in the strong sense even if all setup times are equal to zero, the batch sizes are unbounded, and the function $G(g, f, r)$ is independent of g.*

Proof. We use a reduction from the strongly NP-complete problem 3-Partition.

3-Partition: Given $3q + 1$ positive integer numbers $h_1, \ldots, h_{3q}$ and H such that $H/4 < h_i < H/2$, $i = 1, \ldots, 3q$, and $\sum_{i=1}^{3q} h_i = qH$, is there a partition of the set $\{1, \ldots, 3q\}$ into q disjoint sets $X_1, \ldots, X_q$ such that $\sum_{i \in X_l} h_i = H$ for $l = 1, \ldots, q$?

Given an instance of 3-Partition, we construct the following instance of the problem *Decide-Deadlock*. There are $3q + 1$ products: $3q$ *partition* products $f = 1, \ldots, 3q$, and one *enforcer* product denoted as E. The demand of each partition product is one unit, i.e., $n_f = 1$, $f = 1, \ldots, 3q$, and the demand for the enforcer product is $n_E = q + 1$. The processing requirements are $p_f = h_f$, $f = 1, \ldots, 3q$, and $p_E = 1$. The batch sizes are all unbounded.

The function $G(g, f, r)$ is such that operations on the partition products are all non-defective, odd operations on the enforcer product are all defective and even operations on the enforcer product are all non-defective. All the setup times are equal to zero, the buffer capacity is $B = 1$, and the buffer lower time limit is $M = H + 2$. The due dates play no role in the problem *Decide-Deadlock* because there is no constraint related to the due dates. Therefore, they can be chosen arbitrarily. We show that 3-PARTITION has a solution if and only if there exists a feasible schedule for the constructed instance of the problem *Decide-Deadlock*.

Consider an arbitrary feasible schedule for the constructed instance of the problem *Decide-Deadlock*. Observe that there are exactly $q + 1$ non-defective operations on the enforcer product, and the line is idle between the last defective and the last non-defective operation on the enforcer product because otherwise these two last operations would belong to different batches and the very last operation would be the only operation of the last batch, which would be defective due to the definition of the function $G(g, f, r)$. Since the schedule is feasible, there is no idle time anywhere before the last defective operation. Due to the buffer lower time limit $M = H + 2$ and the buffer capacity $B = 1$, the completion times of any two consecutive defective operations on the enforcer product should be at least $H + 2$ time units apart from each other. If there are at least $q + 2$ defective operations of the enforcer product, then the total space to be filled between them by operations of partition products is at least $(q + 1)H$, and there must be idle times. Therefore, there are exactly $q + 1$ defective operations, and, hence, $q + 1$ batches of operations on the enforcer product, each including one defective and one non-defective operation. Further, there are q time intervals of length of at least H before the last defective operation, which should be filled with the operations on the partition products so that there is no idle time. Since the processing times of these operations are equal to h_f, $f = 1, \ldots, 3q$, and $\sum_{f=1}^{3q} h_f = qH$, the q intervals can be filled with no idle time if and only if problem 3-PARTITION has a solution. ■

It follows from Theorem 1 that a modification of the problem $P(L_{\max})$ in which $L_{\max}$ is replaced by any other objective function, or even if there is no objective at all is NP-hard in the strong sense.

The proof of Theorem 1 can be adapted for the buffer capacity $B > 1$. Specifically, we can assume that the function $G(g, f, r)$ is such that B first operations on the enforcer product are all defective and $(B+1)$-st operations on the enforcer product are all non-defective. The value of B is pseudo-polynomially bounded by q and H. The buffer lower time limit is $M = B + H + 1$. Then, in a feasible schedule, between any two consecutive batches of the enforcer product there must be a gap of at least H time units to fill with operations of the partition products. The remaining logic is the same.

Theorem 2 *The problem No-Deadlock is NP-hard in the strong sense even if all setup times are equal to zero, the batch sizes are unbounded, and the function $G(g, f, r)$ is independent of g.*

Proof. The proof is similar to the previous proof. Given an instance of the problem 3-Partition, we construct the following instance of the problem $P(L_{\max})$, which will be later shown to be an instance of the problem *No-Deadlock*. There are $5q+2$ products. Among them there are *partition* products denoted as $f = 1, \dots, 3q$; *E-enforcer* products denoted as Ej, $j = 1, \dots, q+2$; and *D-enforcer* products denoted as Dj, $j = 1, \dots, q$. All the setup times are equal to zero, the buffer capacity is $B = 1$, and the buffer lower time limit is $M = H+2$. The demand of each product is one unit. The processing requirements are $p_f = h_f$, $f = 1, \dots, 3q$, $p_{Ej} = 1$, $j = 1, \dots, q+2$, and $p_{Dj} = H$, $j = 1, \dots, q$. All the partition products have a common due date $d = (H+2)q$. The due date of the enforcer product Ej is equal to $d_{Ej} = 2+(H+2)(j-1)$, $j = 1, \dots, q+1$. The due date of the enforcer product $E(q+2)$ and the due dates of D-enforcer products are sufficiently large in such that they can never be exceeded. For example, they are equal to $d + \sum_{f=1}^{3q} p_f + 2p_{E(q+1)} + 2p_{E(q+2)} + M = (2H+2)q + H + 6$. The batch sizes are all unbounded. The function $G(g, f, r)$ is such that operations on the partition products and D-enforcer products are all non-defective, odd operations on any E-enforcer product are all defective and even operations on any E-enforcer product are all non-defective.

Observe that the following schedule is feasible for the constructed instance of the problem $P(L_{\max})$. There is a single batch of size two for each E-enforcer product. There is a single batch of size one for each D-enforcer product and each partition product. In the schedule, the first $q+1$ odd batches are the batches of the E-enforcer products in the order $E1, \dots, E(q+1)$, and the first q even batches are batches of the D-enforcer products. The next $3q$ batches are the batches of the partition products, and the last batch is the batch of the E-enforcer product. See Fig. 2 for an illustration.

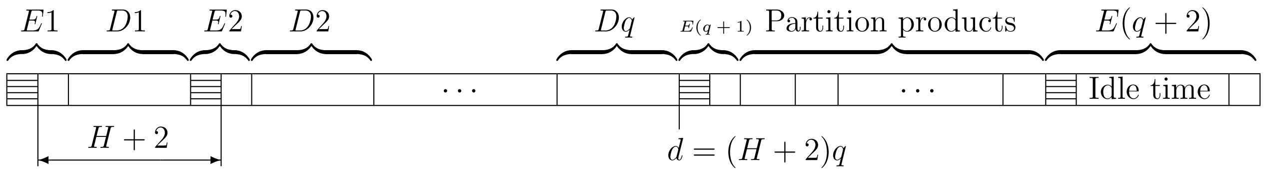

Figure 2: A feasible schedule for the constructed instance of the problem $P(L_{\max})$.

We deduce that the constructed instance is indeed an instance of the problem *No-Deadlock*. We now show that 3-Partition has a solution if and only if there exists a feasible schedule for the constructed instance of the problem *No-Deadlock* with value $L_{\max} \leq 0$.

Consider an arbitrary feasible schedule for the constructed instance of the problem *No-Deadlock* with value $L_{\max} \leq 0$. Similar to the previous proof, observe that there are exactly $q+2$ non-defective operations on the E-enforcer products, and the line is idle between the last defective and the last non-defective operation on a E-enforcer product. Since the schedule is feasible, there is no idle time before the last defective operation. Due to the buffer lower time limit $M = H+2$ and the buffer capacity $B = 1$, the completion times of any two consecutive defective operations on the E-enforcer products should be at least $H+2$ time units apart from each other. Therefore, the $(q+1)$-st defective operation can start no earlier than at time $(H+2)q$, which is the due date for the partition products. Since this due date cannot be exceeded, the partition products should fill completely the q time intervals of length at least H between every two consecutive defective operations in the time period $[0, (H+2)q]$. Since the processing times of these operations are equal to h_f, $f = 1, \ldots, 3q$, and $\sum_{f=1}^{3q} h_f = qH$, the q intervals can be completely filled if and only if problem 3-Partition has a solution. ■

Theorem 2 implies that a modification of the problem $P(L_{\max})$, in which the criterion is to minimize the total number of good quality items produced after the corresponding due dates is NP-hard in the strong sense. Furthermore, it follows from its proof that a modification of the problem $P(L_{\max})$, in which deadlocks are allowed, is NP-hard in the strong sense.

The proof of Theorem 2 can be adapted for the buffer capacity $B > 1$ similar to the adaptation of Theorem 1 for this case.

Theorems 1 and 2 show that an optimal polynomial time solution algorithm for the problem $P(L_{\max})$ is unlikely to exist, and that efficient and practically relevant heuristic procedures are of interest. A heuristic Group Technology (GT) approach for solving the problem $P(L_{\max})$ is analyzed in the following section. This GT approach suggests that a single batch is formed for each product.

4 Group technology solution approach

Group Technology (GT) benefits manufacturing by making process planning easier and more consistent. Furthermore, it reduces setups and tooling cost. This approach is widely used in various scheduling environments, including real car paint shops, see Ham et al. [17], Potts and Van Wassenhove [33], Liaee and Emmons [23] and Solnon et al. [38]. However, no result on its efficiency and limitations is reported in the literature. In this section, we will analyze this aspect.

Let us renumber products in the *Earliest Due Date (EDD)* order such that $d_1 \leq \cdots \leq d_F$. Consider a heuristic solution for the problem $P(L_{\max})$, in which a single batch is formed for each product, every defective item of the same product leaves the buffer at the earliest possible time, and the products are sequenced in the EDD order $(1, \ldots, F)$ with ties broken arbitrarily. We denote such a solution as a *GT-EDD schedule*. It can be constructed in $O(F \log F)$ time, which includes sorting the due dates, an instruction on the continuous production of the same product, and an instruction on handling items in the buffer. The mentioned instructions require $O(1)$ time.

Note that the GT-EDD schedule may be infeasible because a deadlock may occur for it. In the following subsection we will establish conditions under which a GT-EDD schedule is optimal for the problem $P(L_{\max})$.

4.1 Sufficient conditions for optimality of a GT-EDD schedule

It is convenient to introduce some additional notation.

Consider functions $\Delta_f(g,h) := \sum_{r=1}^{h} G(g,f,r)$, $f = 1, \ldots, F$, of variables g and h, $g = 0, 1, \ldots, F$, $h \in \{1, \ldots, U_f\}$. $\Delta_f(g,h)$ is the number of defective operations among first h operations of a batch of product f preceded by a batch of product g. We say that function

$\Delta_f(g,h)$ has a *concave staircase structure* in h if the lengths of intervals of h, in which this function has the same value, non-decrease as h increases with a possible exception for the last interval.

The remaining notation concerns function $\Delta_f(g,h)$ with fixed variable $g = f-1$. Recall that product $f-1$ precedes product f in the GT-EDD schedule. An example of function $\Delta_f(f-1,h)$ with the concave staircase structure is given in Fig. 3. This figure also illustrates the remaining notation.

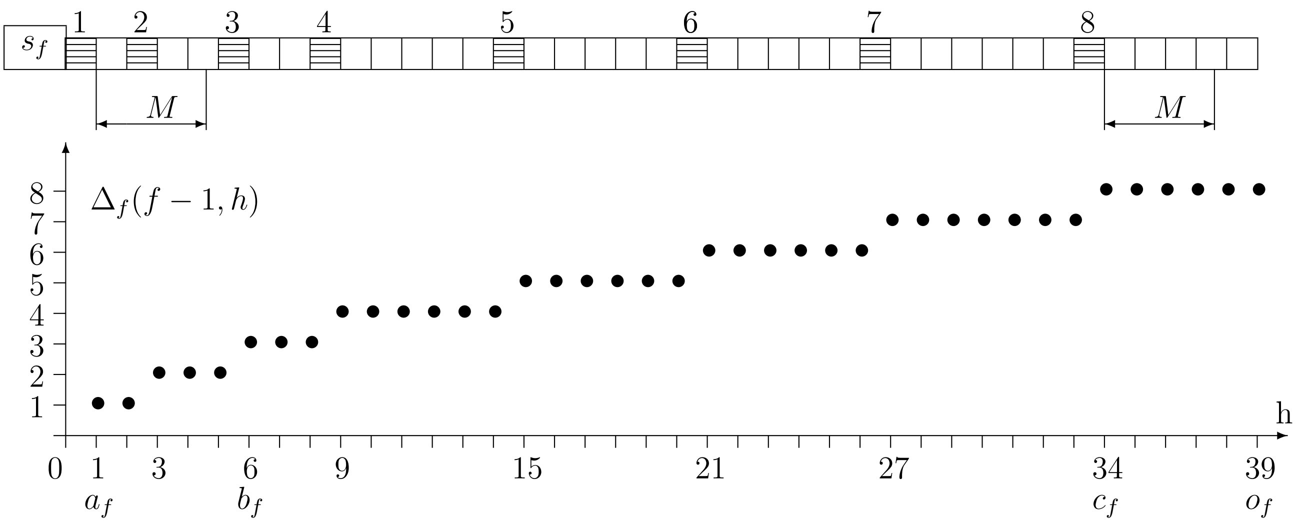

Figure 3: Function $\Delta_f(f-1,h)$ with concave staircase structure in h. Here $n_f = 31$, $a_f = 1$, $b_f = 6$, $c_f = 34$ and $o_f = 39$.

Consider a batch of product f preceded by a batch of product $f-1$. Let the batch of product f contain n_f non-defective operations. Denote by o_f the total number of operations in this batch. It is the unique solution of the equation $o_f - \Delta_f(f-1,o_f) = n_f$. Since the last operation is non-defective, $G(f-1,f,o_f) = 0$. Denote by a_f the position of the earliest defective operation in this batch, $a_f = \min\{h \mid G(f-1,f,h) = 1\}$. Define $a_f = 0$ if there is no defective operation. Denote by b_f the earliest position in this batch, where the first defective item can be reworked, $b_f = a_f + 1 + \lceil \frac{M}{p_f} \rceil$. Denote by c_f the position of the latest defective operation in this batch, $c_f = \max\{h \mid h \le o_f, G(f-1,f,h) = 1\}$.

Theorem 3 *The GT-EDD schedule is optimal for the problem $P(L_{\max})$ if the following conditions (i)-(vi) are satisfied.*

(i) each function $\Delta_f(g,h)$ is minimized in g at $g = f-1$ for any $h \in \{1,\ldots,U_f\}$.

(ii) *each function* $\Delta_f(g,h)$ *has the concave staircase structure in* h.

(iii) $\Delta_f(f-1, b_f-1) \leq B+1,\ f=1,\ldots,F$. For the example in Fig. 3 this condition is satisfied if $B \geq 2$.

(iv) *if* $G(f-1,f,r) = 1$ *for at least one* r, $1 \leq r \leq o_f$, *then* $M \leq p_f(o_f - c_f - 1)$, $f = 1,\ldots,F$. For the example in Fig. 3 this condition is satisfied.

(v) *if* $G(f-1,f,r) = 1$ *for at least one* r, $1 \leq r \leq o_f$, *then* $V_f \geq a_f$, $f = 1,\ldots,F$.

(vi) $V_f \leq o_f \leq U_f$, $f = 1,\ldots,F$.

Proof. Condition (i) ensures that the EDD sequence of products is the best with respect to minimizing the number of defective operations. Condition (ii) assures that the increase of the number of defective operations becomes slower as the batch size becomes larger. Condition (iii) guarantees that the buffer capacity B is large enough to accept defective items without deadlock until the first defective item goes for rework. Condition (iv) ensures that if there is at least one defective operation, then the lower buffer time limit M is small enough to rework the last defective item without deadlock. Condition (v) guarantees that if there is at least one defective operation, then the size of any feasible batch must be large enough to include the first defective operation. Otherwise, schedules with small batches can have no defective operations and they can be the only optimal ones. Condition (vi) requires batches of the GT-EDD schedule to have feasible sizes.

Consider an optimal schedule S^* for problem $P(L_{\max})$ with the objective value $L^*_{\max}$. Let S^* contain x_f defective operations on product f, $f = 1,\ldots,F$. Now consider a batch scheduling problem, which differs from the problem $P(L_{\max})$ in that each product f consists of at most $n_f + x_f$ items, $f = 1,\ldots,F$, and the production is perfect such that every manufactured item is of good quality. We denote this problem as $Perfect$. Santos [36] proved that an optimal solution of problem $Perfect$ exists, in which no product is split into batches, and products are sequenced in the EDD order, see also Potts and Van Wassenhove [33]. Since schedule S^* is feasible for problem $Perfect$, the optimal solution value of this problem, $L^{(0)}_{\max}$, is a lower bound for $L^*_{\max}$: $L^{(0)}_{\max} \leq L^*_{\max}$.

Let the GT-EDD schedule contain y_f defective operations of product f, $f = 1,\ldots,F$. Assume that conditions (i)-(vi) are satisfied. Firstly, we will show that $y_f \leq x_f$, $f = 1,\ldots,F$.

Let there be q_f batches of product f in the optimal schedule S^*, and let j-th batch of product f consist of $o_f^{(j)}$ work and rework operations, among which there are $x_f^{(j)}$ defective operations. Consider an artificial schedule which differs from the GT-EDD schedule in that the single batch of product f consists of $\sum_{j=1}^{q_f} o_f^{(j)}$ work and rework operations, $f = 1, \ldots, F$. Let $z_f^{(j)}$ denote the number of defective operations among operations in positions $\sum_{k=1}^{j-1} o_f^{(k)} + 1, \sum_{k=1}^{j-1} o_f^{(k)} + 2, \ldots, \sum_{k=1}^{j} o_f^{(k)}$, $o_f^{(0)} = 0$, and let z_f denote the total number of defective operations on product f in this artificial GT-EDD schedule. Due to the properties (i), (ii) and (v), we have $\sum_{k=1}^{j} z_f^{(k)} \leq \sum_{k=1}^{j} x_f^{(k)}$, $j = 1, \ldots, q_f$, which implies $z_f \leq x_f$, $f = 1, \ldots, F$. As the artificial GT-EDD schedule has the same total number of operations, it has the same or smaller number of defective operations and the same or larger number of non-defective operations on each product f in comparison with the schedule S^*. Since the GT-EDD schedule and the artificial GT-EDD schedule have the same sequence of operations on product f up to the operation corresponding to the production of n_f-th good quality item of this product, $f = 1, \ldots, F$, we deduce that $y_f \leq z_f$, and, hence, $y_f \leq x_f$, $f = 1, \ldots, F$. It follows from the above discussion that the optimal solution value, $L_{\max}^{(1)}$, of the problem $Perfect$, in which each product f consists of $n_f + y_f$ items, is a lower bound for $L_{\max}^*$: $L_{\max}^{(1)} \leq L_{\max}^*$.

It remains to show that there is no idle time in the GT-EDD schedule. If so, then it coincides with the optimal solution of the problem $Perfect$, in which each product f consists of $n_f + y_f$ items, and, therefore, has value $L_{\max}^{(1)}$. The no idle time property of the GT-EDD schedule is guaranteed by the conditions (ii), (iii), (iv) and (vi). Condition (vi) guarantees that the batch sizes in the GT-EDD schedule are all feasible. Provided that the buffer is empty when an execution of the batch of product f starts, condition (iii) ensures that the buffer capacity will not be exceeded until the earliest time when the first defective item of this product can leave the buffer. Due to the condition (ii), this statement is also satisfied for any defective item. Since defective items leave the buffer at the earliest possible times and the buffer capacity is never exceeded, condition (iv) guarantees that all the operations of the same batch can be executed with no idle time so that any defective item can leave the buffer at a time when an operation on another item is completed. Furthermore, at the end of the product f execution, the buffer will be empty. Thus, the GT-EDD schedule contains no idle time, which concludes the proof. ■

4.2 Examples of non-optimal GT-EDD schedules if one of conditions (i)-(vi) fails

We will show that if one of the conditions (i)-(vi) is violated and the remaining conditions are satisfied, an instance of the problem $P(L_{\max})$ exists for which no GT-EDD schedule is optimal. This characteristic is important but it does not mean that (i)-(vi) are necessary conditions of the existence of an optimal GT-EDD schedule for any given instance because there may exist an instance, for which some of these conditions are violated but the GT-EDD schedule is optimal. In all instances given below there are two products, $s_1 = s_2 = 0$ and $p_1 = p_2 = 1$. In instances illustrating conditions (i)-(v), batch sizes are unbounded.

Assume that condition (i) fails, $n_1 = n_2 = 1$, $d_1 = 2$, $d_2 = 3$, $M = 0$, $B = 1$, $G(0,1,1) = G(0,2,1) = G(2,1,1) = G(1,2,3) = 0$ and $G(1,2,1) = G(1,2,2) = 1$. An optimal schedule and the unique GT-EDD schedule are given in Fig. 4.

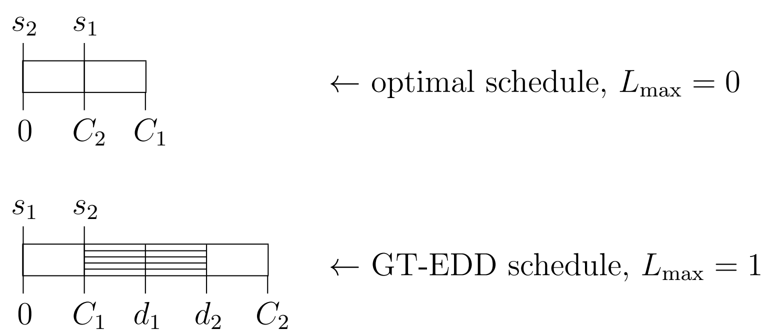

Figure 4: Condition (i) fails: $\Delta_2(1,1) = 1 > \Delta_2(0,1) = 0$.

Assume that condition (ii) fails, $n_1 = 1$, $n_2 = 2$, $d_1 = 3$, $d_2 = 5$, $M = 0$, $B = 1$, $G(0,1,1) = G(2,1,1) = G(0,2,2) = G(1,2,2) = G(1,2,5) = 0$ and $G(0,2,1) = G(1,2,1) = G(1,2,3) = G(1,2,4) = 1$. This example is illustrated by Fig. 5.

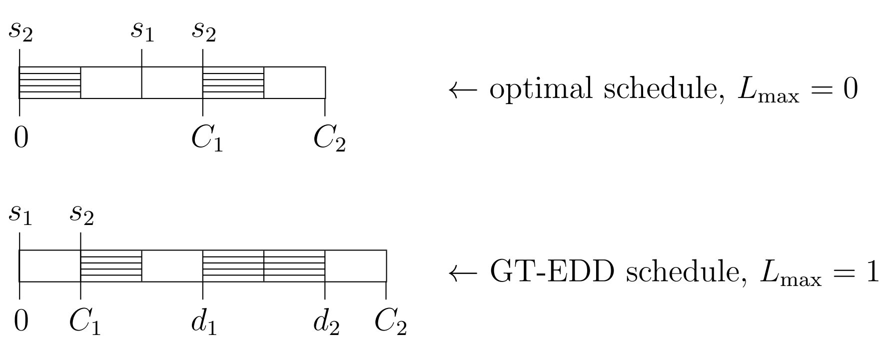

Figure 5: Condition (ii) fails: $\Delta_2(1,h)$ does not have concave staircase structure in h.

Assume that condition (iii) fails, $M = 2$, $B = 1$, $n_1 = 1$, $n_2 = 3$, $d_1 = 2$, $d_2 = 3$, $G(0,1,1) = G(1,2,3) = G(1,2,4) = G(1,2,5) = 0$ and $G(0,2,1) = G(0,2,2) = G(1,2,1) = G(1,2,2) = 1$. In this case no feasible schedule exists, including the GT-EDD schedule, see Fig. 6.

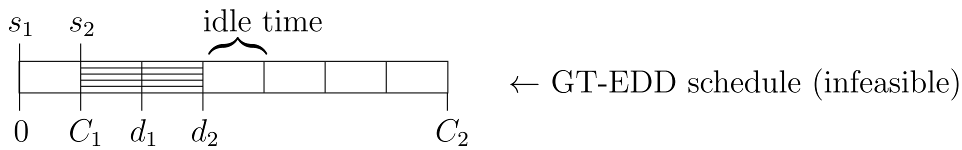

Figure 6: Condition (iii) fails: $\Delta_f(f-1, b_f-2) = \Delta_2(1,2) = 2 > B = 1$.

Assume that condition (iv) fails, $M = 1$, $B = 1$, $n_1 = n_2 = 1$, $d_1 = 1$, $d_2 = 2$, $G(0,2,1) = G(1,2,1) = G(0,1,2) = 0$ and $G(0,1,1) = G(2,1,1) = 1$. In this case, the GT-EDD schedule is infeasible and a feasible schedule exists, see Fig. 7.

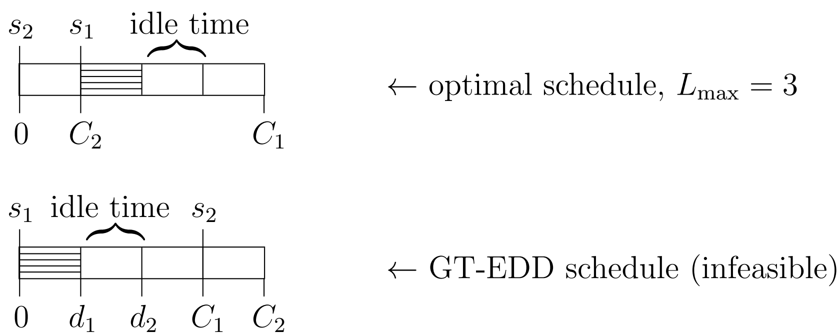

Figure 7: Condition (iv) fails: $M = 1 > p_1(o_1 - c_1 - 1) = 0$.

Assume that condition (v) fails, $M = 0$, $B = 1$, $n_1 = 1$, $n_2 = 2$, $d_1 = 2$, $d_2 = 3$, $G(0,1,1) = G(0,2,1) = G(1,2,1) = G(1,2,3) = 0$ and $G(1,2,2) = 1$. An optimal schedule and the GT-EDD schedule are given in Fig. 8.

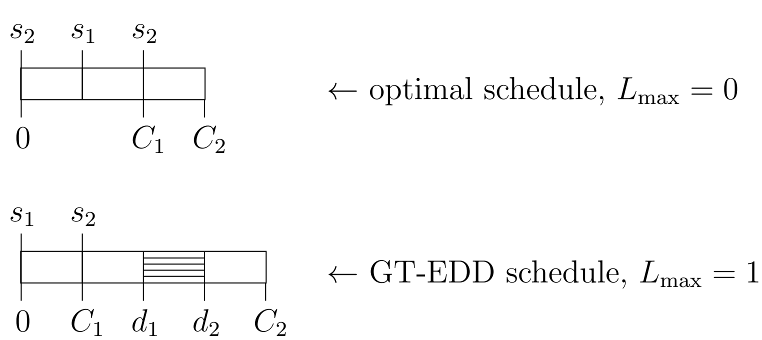

Figure 8: Condition (v) fails: $V_2 = 1 < a_2 = 2$.

Finally, assume that condition (vi) fails. In this case, the GT-EDD schedule is infeasible because at least one of its batch sizes violates the lower or upper bound.

4.3 Practical relevance and managerial implications

Conditions (i)-(vi) are satisfied for the following situations.

- The number of defective operations is smaller if the production switches from a lighter color to the most similar darker color, and it is decided or negotiated with the customers that the products with lighter colors have earlier due dates (condition (i)).

  It should be noted that this condition is difficult to satisfy for the production with a large demand for various colors and competing delivery dates.

- The probability that an operation is defective decreases as its position in the same batch increases because the dissolving, drying and evaporation processes gradually reduce the impact of the previous color (condition (ii)).

- The buffer capacity is sufficiently large and the buffer lower time limit is sufficiently small to avoid a deadlock in the GT-EDD schedule (condition (iii)).

- The buffer capacity is sufficiently large and the buffer lower time limit is sufficiently small such that no idle time occurs and the buffer is empty at the completion time of any product in the GT-EDD schedule (condition (iv)).

- If the batch of any product in the GT-EDD schedule contains at least one defective operation, then every batch of this product in any schedule does (condition (v)).

- The batch sizes in the GT-EDD schedule are feasible (condition (vi)).

Managerial implications of the results in this section are twofold:

1. Conditions (i)-(vi) can be used to make a decision of implementing a GT-EDD schedule in a production environment with buffered rework.

2. Conditions (i)-(vi) can be used to make a decision of modifying a production environment so that the GT-EDD schedule provides good results. For example, it can be decided or negotiated that products with lighter colors have earlier due dates, the buffer's

lower time limit decreases by installing a more efficient drying device, or the buffer is rebuilt to have larger capacity. All these changes increase chances that the GT-EDD schedule minimizes $L_{\max}$, which is good for coordinating decisions in the corresponding supply and production chain.

5 Numerical study

As shown in the previous section, a GT-EDD schedule is optimal if certain realistic assumptions are satisfied. In this section we will analyze the quality of GT-EDD schedules on randomly generated test instances. We restrict ourselves to instances for which a feasible schedule always exists, i.e., instances of the problem *No-Deadlock*.

Note that GT-EDD schedules might be infeasible even for a *No-Deadlock* instance. In order to obtain comparable results, we slightly generalize the GT-EDD heuristic for the infeasible case of $o_f > U_f$: we split the only large batch of product f into several batches of the same size U_f and, possibly, one batch of size smaller than U_f, and sequence them consecutively in this order.

The results of this generalized GT-EDD heuristic are compared with the results of a branch-and-bound algorithm, if they are available within a limited time, a lower bound, and the results of two meta-heuristics, which are based on a simple neighborhood structure. The branch-and-bound algorithm can be characterized as follows. A global upper bound is obtained by applying tabu search, see below. The original problem constitutes the root node, from which branches emanate, each of which stands for a fixed first batch of a certain size. Note that each branch describes a single instance of $P(L_{\max})$ with a reduced demand of the product of the first batch. Therefore, a lower bound of $P(L_{\max})$ can easily be applied to each branch. A lower bound is obtained by setting the buffer lower time limit M to zero, and the buffer capacity B to a sufficiently high value. Furthermore, we relax the defectiveness of some items such that a function G' with $G'(f,r) := \min_g G(g,f,r)$ is used instead of G. In this case, the production time of all items of a product is not influenced by the other products. Thus, the products can be sorted according to the EDD rule. An optimal schedule for a single product f in this relaxed problem minimizes the total setup and processing time, and it can be determined inductively in polynomial time as follows. Firstly, if $n_f = 1$, then it is optimal to have one batch of size $\min\{r|G'(f,r) = 0\}$. Secondly, if $n_f > 1$, then the non-defective items can be produced in one batch or in at least two batches, each of the latter including at most $n_f - 1$ non-defective items. Therefore, we can determine an optimal solution to obtain n_f non-defective items inductively by choosing the best solution among the one-batch schedule with n_f items and all schedules obtained from concatenation of an optimal solution for x and an optimal solution for $n_f - x$ non-defective items, where $x \in \{1, \ldots, \lfloor n_f/2 \rfloor\}$.

The meta-heuristics use the following neighborhood. For a given feasible schedule s, a neighbor is a feasible schedule s' that can be derived from s by performing both of the following modifications exactly once:

1. Decrease the number of operations in one batch such that the number of non-defective operations is reduced by one (this may lead to removal of a batch).

2. Increase the number of operations in one (possibly non-existing) batch such that the number of non-defective operations is raised by one.

Using this neighborhood, we apply a simple *hill climbing* (HC) algorithm, cf. Michalewicz and Fogel [27]. Starting from the GT-EDD solution, the neighborhood is searched for a better solution. If there is no such solution, the algorithm stops. Otherwise, the best solution found is used as current solution, and the algorithm starts over again by exploring the neighborhood of the new solution.

In order to overcome the problem of being trapped in a local minimum, the solution obtained by the HC algorithm is improved by the *tabu search (TS)*, see Glover and Laguna [14]. The TS procedure differs from the HC algorithm in three ways. Firstly, it does not stop even if no improved solution is found, i.e., the objective function value of a current solution might be temporarily worsened. Secondly, each current solution is marked "tabu" for a given number of iterations, the so called "tabu length". Thirdly, not the whole neighborhood is explored, but only non-tabu solutions of the neighborhood.

Besides the neighborhood, TS requires tabu length and a termination criterion to be specified. We have chosen 5 for the tabu length, and 10 iterations without improvement for the termination criterion.

5.1 Test set

We use various values for the problem parameters such that a big variety of instances is obtained. An instance of the problem $P(L_{\max})$ can be described by the number of products F, the buffer lower time limit M, the buffer capacity B, and by the attributes of each product. The latter comprises for each product f the quantity n_f, the processing time p_f, the setup time s_f, the due date d_f, the lower and upper bounds V_f and U_f on the batch size, and the function $G(g, f, r)$ for each product g and each r with $V_f \leq r \leq U_f$. For these parameters, the following values were chosen.

1. Instances with $F = 5$, 10, 15, and 20 products were generated (4 choices).

2. The minimum dwell time M within the buffer was set to 0, 3, 6 and 9 (4 choices).

3. Buffer capacity B was set sufficiently large, specifically, $B = 100$. A large buffer capacity in conjunction with some other parameter settings to be described below ensures that for all the instances the *No-Deadlock* condition is satisfied (1 choice).

4. The range for the number of good quality items of each product is set low, medium or high. In these cases, n_f is randomly drawn (with uniform distribution) from the sets $\{1, \ldots, 4\}$, $\{1, \ldots, 8\}$ and $\{1, \ldots, 12\}$, respectively (3 choices).

5. The processing time p_f is either fixed to 3 for all the products, or it is randomly drawn from the set $\{2, 3, 4\}$ or from the set $\{1, \ldots, 5\}$ (3 choices). The setup time s_f is randomly taken from one of the three sets $\{1, 2, 3\}$, $\{2, 3, 4\}$ and $\{3, 4, 5\}$. Thus, the expected value of the setup times is either lower, equivalent or higher than the expected value of the processing times, which is always 3 (3 choices).

6. Concerning the due dates, we restrict ourselves to the makespan minimization, which is achieved by setting $d_f = 0$ for all $f = 1, \ldots, F$ (1 choice).

7. In order to avoid difficult feasibility checks, there is no restriction on the minimum batch size, i.e., $V_f = 1$ for all $f = 1, \ldots, F$ (1 choice).

8. The maximum batch size is classified as very tightly restricted ($U_f = 3$), tightly restricted ($U_f = 7$), decently restricted ($U_f = 12$), loosely restricted ($U_f = 18$), or unrestricted ($U_f = \infty$) (5 choices).

9. Function $G(g, f, r)$ is assumed to be independent of g, and we distinguish between three cases. Firstly, $\Delta_f(g, h)$ has the concave staircase structure. Secondly, for each $G(g, f, r)$ the corresponding value is randomly drawn. Thirdly, $\Delta_f(g, h)$ has a convex staircase structure, which means that $\Delta_f(g, U_f - h)$ has a concave staircase structure (3 choices).

   The procedure of randomly generating a concave staircase structure or a convex staircase structure can be described as follows.

   The values of $G(g, f, r)$ for $r = 1, \ldots, U_f$ can be represented by a list with U_f elements, each of which is either 0 or 1. The number of defective operations in a maximal batch, $def := \sum_{r=1}^{U_f} G(g, f, r)$ is randomly drawn from $\{0, \ldots, U_f - 1\}$. Start with a list with def elements, which are all equal to 1. Randomly insert a zero element to the list such that the concave (convex) staircase structure is not violated and repeat this inserting step until the list has U_f elements (the first 0 is added to the end of the list, the second 0 is inserted before or behind the last 1, etc.).

We generated the test set using a full factorial design of all the parameter choices and repeated each setting 5 times. Thus, the test set consists of $4 \cdot 4 \cdot 1 \cdot 3 \cdot 3 \cdot 3 \cdot 1 \cdot 1 \cdot 5 \cdot 3 \cdot 5 = 32400$ instances.

Note that for some instances of this test set the generalized GT-EDD schedule might be infeasible. However, this case is rather rare - the GT-EDD schedule was feasible for 30443 out of 32400 instances.

If the generalized GT-EDD schedule is infeasible, it is replaced by a *feasible start (FeaS)* solution, which is obtained by a modification of the generalized GT-EDD heuristic such that defective items are not reworked as long as there are items left in the initial inventory.

5.2 Results

All algorithms have been implemented in Java 2 and run on an Intel® Core$^{\text{TM}}$2 Duo, 2.5 GHz PC, with 3.24 GB of memory. For the branch-and-bound algorithm, we defined a time limit of three minutes for instances with up to five products and stopped testing instances with more than five products, as too many instances hit the time limit and runtimes became too large. From the 8100 instances with five products, 6754 were solved exactly within the time limit. On average, the objective function score of the (generalized) GT-EDD solutions was only 3.9% above the optimum. On these instances, the results of Tabu Search were only 0.37 % above optimum and the lower bound was 1.3% below optimum.

In Table 2, for all 32400 instances, the total makespan value (row $C_{\max}$) is listed for the FeaS solutions, the generalized GT-EDD solutions, the hill climbing algorithm solutions, the tabu search solutions, and the lower bound (columns FeaS, GT-EDD, HC, TS, and LB respectively). Furthermore, the relative deviations from the lower bound (row "Relative to LB") and the total runtime (row "runtime in sec.") are given. One can see that the lower bound is close to the generalized GT-EDD solutions. On average, the generalized GT-EDD solutions are 11.89% above the lower bound and 4.36% above the best solution found (TS). The FeaS solutions are on average 20.92% above the lower bound.

	FeaS	GT-EDD	HC	TS	LB
$C_{\max}$	9273721	8581219	8252180	8222843	7669293
Relative to LB	1.2092	1.1189	1.076	1.072	1
Runtime in sec.	1.18	132.52	882.52	3079	.36

Table 2: Total makespan and average deviations of the procedures on 32400 instances.

Note that the the generalized GT-EDD solution cannot be better than that of HC or TS, as it is used as an initial solution in HC and TS algorithms. As for the FeaS solution, it was better than the generalized GT-EDD solution in 1991 cases (6.15%). In 74 cases, the FeaS solution was even better than the solution found by TS.

We also determined the most influencing factors which affect the relative gap between the values of the GT-EDD and the lower bound. The relative gap was calculated for each instance, and the Pearson Product Moment Correlation Coefficient (PPMCC) was determined for the parameters in Table 3.

F	M	n_f	p_f	s_f	U_f
0.04	0.38	0.08	0.10	-0.05	-0.29

Table 3: Correlation of parameters affecting the relative gap (GT-EDD)/(LB).

The influence of the number of products F, the number of items n_f, the processing time p_f, and the setup time s_f appears to be negligible, as the absolute value of the PPMCC was never higher than 0.1. A noticeable influence have the maximum batch size U_f and especially the buffer lower time limit M. The latter is no surprise, as the objective function score increases, with increasing M, but not the lower bound. Yet, there is still a noticeable correlation if GT-EDD is compared to the best solution found (TS). In this case, the PPMCC is 0.25. Let us take a closer look on the two parameters with considerable correlation.

M		FeaS	GT-EDD	HC	TS	LB
0	$C_{\max}$	2181779	1952795	1929815	1926940	1916267
0	Relative to LB	1.1386	1.0191	1.0071	1.0056	1
0	Runtime in sec.	0.45	0.06	119.21	596.34	.09
3	$C_{\max}$	2260589	2072011	2007686	2000140	1918130
3	Relative to LB	1.1785	1.0802	1.0467	1.0428	1
3	Runtime in sec.	0.32	34.71	221.28	796.59	0.1
6	$C_{\max}$	2352572	2195212	2105861	2096349	1915926
6	Relative to LB	1.2279	1.1458	1.0991	1.0942	1
6	Runtime in sec.	0.19	50.62	237.52	808.99	0.1
9	$C_{\max}$	2478781	2361201	2208818	2199414	1918970
9	Relative to LB	1.2917	1.2305	1.1510	1.1461	1
9	Runtime in sec.	0.21	47.12	304.52	877.17	0.08

Table 4: Comparison of the procedures for different values of M.

In Table 4, the results of the four algorithms and the lower bound are presented for different values of M. Most obviously, the average objective function value increases as M increases, which holds true for all the algorithms but not for the lower bound. Furthermore, it points out that the average relative deviation of the generalized GT-EDD solutions increases as M increases. It appears that for the test instances the generalized GT-EDD heuristic delivers solutions which are close to optimum if M is rather small. For instances with $M = 0$ the gap to the lower bound is only 1.91%.

Secondly, the influence of different values U_f is shown in Table 5. Although there are some exceptions, we may state that in general for the constructed instances a higher value of U_f implies a lower objective function value. Contrary to the previous cases, the deviation of GT-EDD from the lower bound (and also to the TS values) does not change in a straightforward manner: it is medium with the value 11.36% for the case of $U_f = 3$, then it increases up to 17.68% for $U_f = 7$ and falls down to 0.22% if the batch size is not restricted at all.

U_f		FeaS	GT-EDD	HC	TS	LB
3	$C_{\max}$	2355811	2102395	2071958	2071714	1887967
3	Relative to TS	1.2478	1.1136	1.09745	1.0973	1
3	Runtime in sec.	0.14	37.08	357.36	1558.85	0.1
7	$C_{\max}$	2426978	2164579	2031186	2026704	1839421
7	Relative to TS	1.3194	1.1768	1.1043	1.1018	1
7	Runtime in sec.	0.33	58.96	353.39	963.70	0.08
12	$C_{\max}$	1909715	1775780	1665712	1653243	1537302
12	Relative to TS	1.2423	1.1551	1.0835	1.0754	1
12	Runtime in sec.	0.23	31.42	140.57	387.90	0.06
18	$C_{\max}$	1465657	1424656	1370474	1358754	1293205
18	Relative to TS	1.1334	1.1017	1.0598	1.0507	1
18	Runtime in sec.	0.19	4.98	27.07	128.74	0.08
∞	Sum	1115560	1113809	1112850	1112428	1111398
∞	Relative to TS	1.0037	1.0022	1.0013	1.0009	1
∞	Runtime in sec.	0.29	0.08	4.12	39.89	0.05

Table 5: Comparison of the procedures for different values of U_f.

We may conjecture that the quality of the easily applicable GT-EDD solutions is very close to the optimum. This conjecture is especially reasonable if M is low and if U_f is either very low or very high. If we only consider instances with $M = 0$ and $U_f = 3$ ($U_f = \infty$), the deviation of GT-EDD from the lower bound is only 0.59% (0.07%).

6 Final remarks

We have studied a problem of scheduling work and rework processes on a single facility with buffered rework, $P(L_{\max})$. The problem is motivated by the optimal scheduling decisions in a car paint shop. The specificity of the problem is that the production is essentially discrete, the defective items are stored in a buffer of a limited capacity, a lower bound on the storage time is given, there are product dependent setup times, no deterioration occurs to the defective items, and the objective is to minimize the maximum lateness of the product demand satisfaction times with respect to their given due dates. The defectiveness of an item is determined by a given simulated function of three variables: the product of this item, the preceding product in the manufacturing/remanufacturing sequence, and the position of an operation on this item in its batch. An optimal search is limited to schedules which contain no deadlock. The deadlock is a situation when the buffer is full, a defective item blocks the line, and there is an item to be manufactured but it cannot because the line is blocked. The problem is proved to be NP-hard in the strong sense for two special cases in which the existence of a deadlock is unknown and known, respectively. A heuristic Group Technology (GT) solution approach is suggested. It constructs the GT-EDD schedule, in which there is a single batch for each product, the products are sequenced in the Earliest Due Date (EDD) order, and defective items leave the buffer as soon as possible. Sufficient conditions for the GT-EDD schedule to be an optimal solution for the studied problem are established. These conditions justify the application of the GT solutions in scheduling car paint shops with buffered rework. Beyond that, the effectiveness of the GT-EDD schedule in general, i.e., in a situation where the optimality conditions are not necessarily fulfilled, is demonstrated by a computational experiment. For the randomly generated data, the values of the GT-EDD solutions stay in a small range compared to those of the solutions of a branch-and-bound algorithm or a lower bound, respectively.

Problem $P(L_{\max})$ and the obtained results can be generalized to the stochastic case as follows. Function $G(g, f, r)$ will be re-defined as the probability that an operation in position r of the current batch of product f preceded by a batch of product g is defective. Then C_f, $L_f = C_f - d_f$, and $L_{\max} = \max\{L_f \mid f = 1, \ldots, F\}$ will represent the expected completion time of the last good quality item of product f, the expected lateness of product f, and the expected maximum lateness of products, respectively, in a given schedule. Bounds B, M, V_f and U_f will limit corresponding expected values.

Stochastic versions of the problems *Decide-Deadlock* and *No-Deadlock* will remain NP-hard in the strong sense because in Theorems 1 and 2 values of $G(g, f, r)$ can be treated as probabilities. Regarding the analysis of the Group Technology approach, modifications of the sufficient optimality conditions (i)-(vi) are not so straightforward. We consider them as the topic for future research. For future research, it is also interesting to study problems with the following features.

- The objective is to minimize the number of setups.
- Various strategies for emptying the buffer.
- Various layouts of the buffer.

References

[1] Agnetis, A., Hall, N.G., Pacciarelli, D. (2006) Supply chain scheduling: Sequence coordination, *Discrete Applied Mathematics* **154** 2044-2063.

[2] Allahverdi, A., Ng, C.T., Cheng, T.C.E., Kovalyov, M.Y. (2008) A survey of scheduling problems with setup times or costs, *European Journal of Operational Research* **187** 985-1032.

[3] Bonsma, P., Epping, Th., Hochstättler, W. (2006) Complexity results on restricted instances of a paint shop problem for words, *Discrete Applied Mathematics* **154** 1335-1343.

[4] Boysen, N., Fliedner, M., Scholl, A. (2009) Sequencing mixed-model assembly lines: Survey, classification and model critique, *European Journal of Operational Research* **192** 349-373.

[5] Buscher, U., Lindner, G. (2007) Optimizing a production system with rework and equal sized batch shipments, *Computers and Operations Research* **34** 515-535.

[6] Chiu, S.W. (2010) Robust planning in optimization for production system subject to random machine breakdown and failure in rework, *Computers and Operations Research* **37** 899-908.

[7] Cordeau, J.-F., Laporte, G., Pasin F. (2008) Iterated tabu search for the car sequencing problem, *European Journal of Operational Research* **191** 945-956.

[8] de Brito, M., Dekker, R. (2004) Reverse logistics: a framework. In: Dekker R., Fleischmann M., Inderfurth K. and van Wassenhove L. N. (eds.), *Reverse Logistics - Quantitative Models for Closed-Loop Supply Chains*, Springer, 3-27.

[9] Epping, Th., Hochstättler, W., Oertel, P. (2004) Complexity results on a paint shop problem, *Discrete Applied Mathematics* **136** 217-226.

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

[11] Gagné, C., Gravel, M., Price, W.L. (2006) Solving real car sequencing problems with ant colony optimization, *European Journal of Operational Research* **174** 1427-1448.

[12] Giard, V., Jeunet, J. (2010) Optimal sequencing of mixed models with sequence-dependent setups and utility workers on an assembly line, *International Journal of Production Economics* **123** 290-300.

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

[14] Glover, F., Laguna, M. (1997) *Tabu Search*, Kluwer, Norwell, MA.

[15] Golle, U., Boysen, N., Rothlauf, F. (2010) Analysis and design of sequencing rules for car sequencing, *European Journal of Operational Research* **206** 579-585.

[16] Guerre-Chaley, F., Frein, Y., Bouffard-Vercelli, R. (1995) An efficient procedure for solving a car sequencing problem. In: INRIA/IEEE Symposium on Emerging Technologies and Factory Automation **2** 385-393.

[17] Ham, I., Hitomy, K., Yoshida, T. (1985) Group technology: applications to production management, Kluwer-Nijhoff, Boston.

[18] Holweg, M. (2002) Product variety - friend or foe? Technical Report, Lean Enterprise Research Centre, Cardiff Business School, United Kingdom.

[19] Inderfurth, K., Janiak, A., Kovalyov, M.Y., Werner, F. (2006) Batching work and rework processes with limited deterioration of reworkables, *Computers and Operations Research*, **33** 1595-1605.

[20] Inderfurth, K., Kovalyov, M.Y., Ng, C.T., Werner, F. (2007) Cost minimizing scheduling of work and rework processes on a single facility under deterioration of reworkables, *International Journal of Production Economics*, **105** 345-356.

[21] Inderfurth, K., Teunter, R.H. (2003) Production planning and control of closed-loop supply chains. In: Guide Jr. V.D.R and van Wassenhove L.N. (eds.), *Business perspectives on closed-loop supply chains*, Carnegie Mellon University Press, 149-173.

[22] Joly, A., Frein, Y. (2008) Heuristics for an industrial car sequencing problem considering paint and assembly shop objectives, *Computers and Industrial Engineering* **55** 295-310.

[23] Liaee, M.M., Emmons, H. (1997) Scheduling families of jobs with setup times, *International Journal of Production Economics* **51** 165-176.

[24] Liu, J.J., Yang, P. (1996) Optimal lot sizing in an imperfect production system with homogeneous reworkable jobs, *European Journal of Operational Research* **91** 517-527.

[25] Meunier, F., Seb, A. (2009) Paintshop, odd cycles and necklace splitting, *Discrete Applied Mathematics* **157**, 780-793.

[26] Meyr, H. (2004) Supply chain planning in the German automotive industry, *OR Spectrum* **26** 447-470.

[27] Michalewicz, Z., Fogel, D.B. (2004) How to solve it: Modern heuristics, Springer, Berlin.

[28] Morin, S., Gagne, C., Gravel, M. (2009) Ant colony optimization with a specialized pheromone trail for the car-sequencing problem, *European Journal of Operational Research* **197** 1185-1191.

[29] Pawlak, G., Lichocki, P., Plaza, M. (2006) On-line algorithms for minimizing the number of colour changes in a car paint shop buffer. In: *PMS 2006, Tenth International Workshop on Project Management and Scheduling, April 26-28, 2006, Poznan, Poland. Abstracts*, Wydawnictwo Nakom, Poznan, 275-280.

[30] Pawlak, G., Mamoński, M., Sobkowski, P. (2006) Assembly line balancing in the buffer of the car production line. In: *PMS 2006, Tenth International Workshop on Project Management and Scheduling, April 26-28, 2006, Poznan, Poland. Abstracts*, Wydawnictwo Nakom, Poznan, 281-285.

[31] Pawlak, G., Rucinski, M. (2006) On-line algorithm for minimizing the makespan in the re-entrant single machine system. In: *PMS 2006, Tenth International Workshop on Project Management and Scheduling, April 26-28, 2006, Poznan, Poland. Abstracts*, Wydawnictwo Nakom, Poznan, 286-291.

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

[33] Potts, C.N., Van Wassenhove, L.N. (1992) Integrating scheduling with batching and lot-sizing: a review of algorithms and complexity, *Journal of the Operational Research Society* **43** 395-406.

[34] Ribeiro, C.C., Aloise, D., Noronha, T.F., Rocha, C., Urrutia, S. (2008) A hybrid heuristic for a multi-objective real-life car sequencing problem with painting and assembly line constraints, *European Journal of Operational Research* **191** 981-992.

[35] Rosenblatt, M.J., Lee, H.L. (1986) Economic production cycles with imperfect production processes, *IIE Transactions* **18** 48-55.

[36] Santos, C. (1984) Batching and sequencing decisions under lead time considerations for single machine problems. MSc Thesis, Department of Management Sciences, University of Waterloo, Waterloo.

[37] Sawik, T.J. (1993) A scheduling algorithm for flexible flow lines with limited intermediate buffers, *Applied Stochastic Models and Data Analysis* **9** 127-138.

[38] Solnon, C., Cung, V.D., Nguyen, A., Artigues, C. (2008) The car sequencing problem: Overview of state-of-the-art methods and industrial case-study of the ROADEF2005 challenge problem, *European Journal of Operational Research* **191** 912-927.

[39] Spieckermann, S., Gutenschwager, K., Voss, S. (2004) A sequential ordering problem in automotive paint shops, *International Journal on Production Research* **42** 1865-1878.

[40] Teunter, R.H., 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.

[41] Valckenaers, P., Van Brussel, H. (2003) Deadlock avoidance in flexible flow shops with loops, *Journal of Intelligent Manufacturing* **14** 137-144.

[42] Wang, X., Tang, L. (2009) A tabu search heuristic for the hybrid flowshop scheduling with finite intermediate buffers, *Computers and Operations Research* **36** 907-918.

[43] Wee, H.M., Widyadana, J.A. (2013) A production model for deteriorating items with stochastic preventive maintenance time and rework process with FIFO rule, *Omega* **41** 941-954.

[44] Zinflou, A., Gagne, C., Gravel, M. (2010) Genetic algorithm with hybrid integer linear programming crossover operators for the car-sequencing problem, *INFOR* **48** 23-37.