Jena Research Papers in Business and Economics

Level Scheduling under limited Resequencing Flexibility

Nils Boysen, Malte Fliedner, Armin Scholl

06/2010

Jenaer Schriften zur Wirtschaftswissenschaft

Working and Discussion Paper Series
School of Economics and Business Administration
Friedrich-Schiller-University Jena

ISSN 1864-3108

Publisher:

Wirtschaftswissenschaftliche Fakultät
Friedrich-Schiller-Universität Jena
Carl-Zeiß-Str. 3, D-07743 Jena

www.jbe.uni-jena.de

Editor:

Prof. Dr. Hans-Walter Lorenz
h.w.lorenz@wiwi.uni-jena.de

Prof. Dr. Armin Scholl
armin.scholl@wiwi.uni-jena.de

www.jbe.uni-jena.de

Level Scheduling under limited Resequencing Flexibility

Nils Boysen[a,*], Malte Fliedner[a], Armin Scholl[b]

[a] *Friedrich-Schiller-University Jena, Chair of Operations Management, Carl-Zeiß-Straße 3, D-07743 Jena, Germany,*
{nils.boysen,malte.fliedner}@uni-jena.de

[*] *Corresponding author, phone +49 3641 943100.*

[b] *Friedrich-Schiller-University Jena, Chair of Management Science, Carl-Zeiß-Straße 3, D-07743 Jena, Germany,*
armin.scholl@uni-jena.de

Abstract
A mixed-model assembly line requires the solution of a short-term sequencing problem, which decides on the succession of different models launched down the line. A famous solution approach stemming from the Toyota Production System is the so-called Level Scheduling (LS), which aims to distribute the part consumption induced by a model sequence evenly over the planning horizon. LS attracted a multitude of different researchers, who, however, invariably treat initial sequence planning where all degrees of freedom in assigning models to production cycles exist. In the real-world, conflicting objectives and restrictions of preceding production stages, i.e., body and paint shop, simultaneously need to be considered and perturbations of an initial sequence will regularly occur, so that the sequencing problem often becomes a re-sequencing problem. Here, a given model sequence is to be reshuffled with the help of re-sequencing buffers (denoted as pull-off tables). This paper shows how to adopt famous solution approaches for alternative LS problems, namely the Product-Rate-Variation (PRV) and the Output-Rate-Variation (ORV) problem, if the (re-)assignment of models to cycles is restricted by the given number of pull-off tables. Furthermore, the effect of increasing re-sequencing flexibility is investigated, so that the practitioner receives decision support for buffer dimensioning, and the ability of the PRV in reasonably approximating the more detailed ORV in a re-sequencing environment is tested.

Keywords: Mixed-model assembly line; Just-in-Time; Level Scheduling; Re-Sequencing

1 Introduction

The sequencing of different product models launched down a mixed-model assembly line is a widespread short-term decision problem perpetually arising in industries, which mass-produce customizable products to order, e.g., automobile industry. A famous solution approach for this problem stems from the Toyota Production System and is denoted as Level Scheduling (LS). LS propagates to spread material demands induced by the model sequence evenly over the planning horizon. This way, demand peaks are avoided and just-in-time production and distribution of parts is facilitated, because a steady demand stream allows reducing expensive safety stocks near the line. A detailed discussion of LS is provided in the review papers by Kubiak (1993), Dhamala and Kubiak (2005) as well as Boysen et al. (2009a).

The traditional LS, which was initially developed at Toyota (see Monden, 1998), aims at a leveling of each part's consumption pattern. Kubiak (1993) refers to this case of LS as Output Rate Variation (ORV) problem, because materials constitute the outputs of preceding production levels, whose actual demand rates are to be leveled. Within the ORV problem, each part type receives a target demand rate, which is determined by distributing the material's overall demand evenly over the planning horizon. Then, a sequence is sought where actual demand rates for all parts are as close as possible to the ideal target rates in every production cycle. Figure 1 gives a schematic representation of the basic concept for a single part.

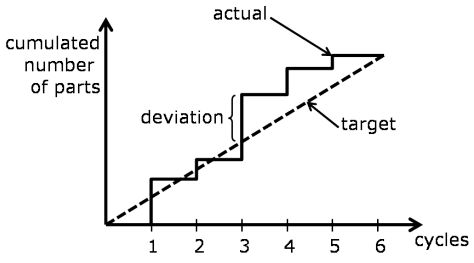

Figure 1: Schematic representation of ORV for a single part

As the ORV was shown to be NP-hard (see Kubiak, 1993, and Kubiak et al., 1997), a simplified LS approach labeled Product Rate Variation (PRV) problem has been introduced (Miltenburg, 1989), which is solvable in pseudo-polynomial runtime (see Kubiak and Sethi, 1991, Steiner and Yeomans, 1993). The more aggregate PRV is claimed to reasonably approximate the ORV (under specific circumstances) without explicitly considering the materials contained in products. Instead, the PRV defines a target production rate for each model type (product), which is then to be approximated by each types's actual production rate.

These alternative versions of the basic LS idea attracted multiple researchers and a recent review paper on sequencing mixed-model assembly lines (Boysen et al., 2009a) lists more than 70 papers on this topic. However, existing research invariably treats initial sequence planning, where no restrictions according to the assignment of models to production cycles exist. In real-

world applications, a re-sequencing of given sequences is often equally essential. On the one hand, workpieces visit multiple departments, i.e., automobile production is subdivided into body and paint shop as well as final assembly. A sequence that is optimal for one department is usually suboptimal for other departments. Consequently, re-sequencing buffers can be applied to reshuffle a sequence according to each's departments individual objective instead of producing an unchanged compromise sequence. On the other hand, disturbances like machine breakdowns, rush orders or material shortages occur with utmost probability, so that initial sequences are stirred up. Especially the paint shop, where smallest defects in color necessitate a retouch or complete repainting of cars, is a widespread reason for disordered model sequences. Again, re-sequencing buffers can be applied to regain desired model sequences before final assembly.

Off-line buffers – also denoted as pull-off tables (see Lahmar et al., 2003) – are a widespread form of organizing re-sequencing buffers. Here, the current on-line model of the initial sequence can be pulled off-line into a free pull-off table, so that successive models can be brought forward and processed before the off-line model is reinserted from its pull-off table back into a later sequence position. Note that each pull-off table is directly accessible. The paper on hand treats the re-sequencing versions of PRV and ORV, where a given number of pull-off tables can be applied to rearrange an initial model sequence, so that material demand is evenly spread over time.

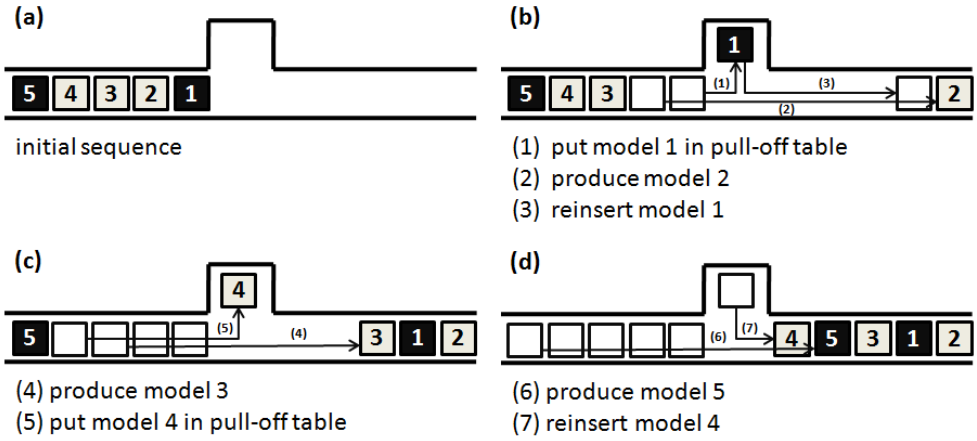

Figure 2: Example for the LS re-sequencing problem with five models and a single pull-off table

Example: An illustrative example is given in Figure 2. Here, a given initial sequence (numbered from model 1 to 5) arrives from a preceding production stage (see Figure 2(a)), where models 1 and 5 (black) as well as models 2, 3 and 4 (grey), respectively, share a common model type. According to the PRV, the models of the different types are to be evenly spread over the five production cycles, where a single pull-off table can be applied to reshuffle the initial sequence. As Figure 2(b) shows models 1 and 2 change positions by pulling model 1 off-line and reinserting it after model 2. Then, model 3 is produced and models 4 and 5 change positions (Figure 2(c)), so that in the final sequence of Figure 2(d) both model types are scheduled in alternating manner

and the resulting sequence is better leveled than the initial one.

Existing research on mixed-model assembly lines does not treat re-sequencing in an LS context. Instead, mainly sequence alterations in front of the paint shop to build larger lots of identical color are investigated, e.g., by Lahmar et al. (2003), Spieckermann et al. (2004), Lahmar and Benjaafar (2007) as well as Lim and Xu (2009). Other contributions for various re-sequencing settings stem from Inman and Schmeling (2003), Ding and Sun (2004) and Gusikhin et al. (2008).

The remainder of the paper is organized as follows. Section 2 treats the adoptions required when re-sequencing is applied in a PRV context. First, the resulting re-sequencing problem is formalized and the exact PRV solution approaches of Kubiak and Sethi (1991) for the sum-squared-deviation function and Steiner and Yeomans (1993) for the max-abs-case are adopted to solve our re-sequencing problem. In a comprehensive computational study, the influence of the buffer capacity (number of pull-off tables) is evaluated, so that the production manager gets some decision support for buffer dimensioning. Re-sequencing in the ORV context is investigated in Section 3. Again, the problem is formalized, a solution approach is presented, which is adapted from the graph approach of Lim and Xu (2009), and the impact of an increasing re-sequencing flexibility is evaluated. Section 4 tests the ability of the re-sequencing version of PRV in approximating the ORV by comparing both LS alternatives in a realistic setting. The final Section 5 concludes the paper.

2 The Product Rate Variation problem under limited re-sequencing flexibility (PRVR)

2.1 Problem description

Consider an initial sequence of models leaving a preceding production stage, e.g., the paint shop, which contains models $i = 1, \ldots, T$ numbered according to their initial sequence position. This initial sequence is to be re-sequenced with the help of a given number K of pull-off tables. Each pull-off table can store a single model at a time and can be accessed directly so that models pulled off-line can be re-inserted in an arbitrary order. Thus, the re-sequencing version of PRV (denoted as PRVR) aims at a mapping $\sigma : \{1, \ldots, T\} \rightarrow \{1, \ldots, T\}$ specifying the reshuffled sequence to be produced in the succeeding production stage, e.g., final assembly. Note that such a mapping ensures that each model receives a unique position in σ and is thus produced exactly once.

Furthermore, for a solution being feasible limited re-sequencing flexibility needs to be considered. In order to shift a model i to an earlier position of the sequence (forward shift), it is necessary to remove preceding models from the sequence and reinsert them after model i has passed. If K pull-off tables are available, a forward shift from a position i to an earlier position $i-1, \ldots, i-K$ is possible. Let $\sigma(i)$ denote the new sequence position of model i in reshuffled

sequence σ, limited re-sequencing flexibility is reflected by condition (1):

$$\sigma(i) \geq \max\{1, i-K\} \quad \forall i = 1, \ldots, T. \tag{1}$$

Each model i is a specific copy of a model type (blueprint) $m \in M$ (with $|M| \leq T$). The number of copies of model type m constitutes its demand d_m. The model type a model i is assigned to is denoted as $a(i)$. It is the aim of PRVR to evenly spread the demand of the model types over the planning horizon. To enable a comparison of (cumulated) actual and target production rates, the number $oc(\sigma, m, t)$ of occurrences of each model type m in a sequence σ up to sequence position t needs to be defined, where $\sigma^{-1}(t)$ denotes the model at position t of reshuffled sequence σ:

$$oc(\sigma, m, t) = |\{\tau \in \{1, \ldots, t\} : a(\sigma^{-1}(\tau)) = m\}| \quad \forall\, m \in M;\, t = 1, \ldots, T \tag{2}$$

Furthermore, a target production rate r_m is to be defined for each model type m by distributing its demand evenly over the planning horizon of length T: $r_m = \frac{d_m}{T}$. With these values on hand, the deviation DEV_{mt} between cumulated actual (first term) and cumulated target (second term) production quantity can be determined in objective function (3):

$$\text{minimize } Z(\sigma) = G\left(F_m\left(DEV_{mt} = oc(\sigma, m, t) - t \cdot r_m\right)\right) \tag{3}$$

Thus, within PRVR a mapping $\sigma : \{1, \ldots, T\} \rightarrow \{1, \ldots, T\}$ is to be determined, which minimizes objective function (3) subject to constraints (1). In the literature, different forms of aggregation functions $G(\cdot)$ and (possibly model type specific) deviation functions $F_m(\cdot)$ have been investigated, which consolidate single deviations DEV_{mt} over all model types m and production cycles t. In the following, we will consider the sum-squared (Section 2.2) and the max-abs-case (Section 2.3) of PRVR.

2.2 Solving the sum-squared-case

The sum-squared-case sums up squared deviations over all cycles t and model types m, so that the objective function takes the following form:

$$\text{minimize } Z^1(\sigma) = \sum_{t=1}^{T} \sum_{m \in M} (DEV_{mt})^2 \tag{4}$$

For the corresponding PRV problem, Kubiak and Sethi (1991, 1994) present an exact solution approach (labeled KS in the following), which is based on the linear assignment problem and runs in $O(T^3)$ time. We will briefly summarize the KS-procedure and show how to adopt it for the sum-squared-version of PRVR.

First, for each model i its ideal production cycle ipc_i, which causes least deviation from target rate for its model type $a(i)$, is to be determined as follows with σ' denoting the initial sequence:

$$ipc_i = \left\lceil \frac{(2 \cdot oc(\sigma', a(i), i) - 1) \cdot T}{2 \cdot d_{a(i)}} \right\rceil \quad \forall\, i = 1, \ldots, T \tag{5}$$

Note that formula (5) takes into account that model i is the $oc(\sigma', a(i), i)$th occurrence of a copy of model type $a(i)$ in the initial sequence. Typically, at least some models compete for identical ideal positions, so that model types need to be coordinated. This is enabled by calculating penalty costs C_{it}, which amount to additional deviations surmounting those resulting from ideal position, if model i is assigned to cycle t:

$$C_{it} = \begin{cases} \sum_{\tau=t}^{ipc_i-1} \left(\left(j - \tau \cdot r_{a(i)}\right)^2 - \left((j-1) - \tau \cdot r_{a(i)}\right)^2 \right), \text{ if } t < ipc_i \\ 0, \text{ if } t = ipc_i \\ \sum_{\tau=ipc_i}^{t-1} \left(\left((j-1) - \tau \cdot r_{a(i)}\right)^2 - \left(j - \tau \cdot r_{a(i)}\right)^2 \right), \text{ if } t > ipc_i \end{cases} \quad \forall\, i, t = 1, \ldots, T \tag{6}$$

These penalty values constitute the cost coefficients of the linear assignment problem to be finally solved within KS. The result is an assignment of models i to production cycles t, which directly constitutes the assembly sequence sought.

The only modification required when solving PRVR with the KS-procedure is to exclude those assignments between models and production cycles in the linear assignment problem, which would lead to an infeasible reshuffled sequence. This can be easily ensured by setting the respective penalty costs of all those arcs, which represent more than K forward position shifts, to a prohibitive value:

$$C_{it} = \infty \quad \forall\, i, t = 1 \ldots, T, \text{ with } i - t > K \tag{7}$$

Example (cont.): In our example of Figure 2, there are two model types (black (type 1) and grey (type 2)), whose target demand rates amount to $r_1 = \frac{2}{5}$ and $r_2 = \frac{3}{5}$, respectively. An optimal sequence for PRV (i.e., PRVP with unlimited number of pull-off tables) with an objective value of $Z^1(\sigma) = 0.8$ is $\sigma = (2, 1, 3, 5, 4)$. Sequence σ can also be attained from the initial sequence $\sigma' = (1, 2, 3, 4, 5)$ with a single pull-off table as depicted in Figure 2.

Clearly, the necessary modifications do not alter computational complexity, so that PRVR can be solved in $O(T^3)$ as is the case with PRV. Interestingly, this means that PRVR can be solved in polynomial time in the length of a reasonably encoded problem instance, since the input length of PRVR directly depends on T (the length of the initial sequence). This is not true for the traditional PRV as the input data only consists of integer model demands, so that T (the sum over all model demands) is not polynomially bounded by their bit size (see Kubiak, 2003 for a more detailed discussion). Note that the KS-procedure also solves the sum-abs-case ($G(F_m(\cdot)) = \sum_{t=1}^{T} \sum_{m \in M} |\cdot|$) to optimality (see Kubiak and Sethi, 1991, 1994), so that the presented extension solves this case of PRVR, as well.

2.3 Solving the max-abs-case

If the maximum absolute deviation over all cycles t and model types m is to be minimized within PRVR, the objective function is defined as follows:

$$\text{minimize } Z^2(\sigma) = \max_{t=1}^{T} \max_{m \in M} |DEV_{mt}| \tag{8}$$

For this case of the PRV, Steiner and Yeomans (1993) introduced an exact solution procedure (denoted as SY), which also solves instances in pseudo-polynomial time ($O(T \log \max_{m \in M}\{d_m\})$). Again, we briefly summarize SY and present the extensions required for solving PRVR.

The SY-procedure reduces the problem to a set of feasibility problems, each of which being initialized with a given maximum deviation DEV. With given DEV for each model i the set Θ_i of feasible production cycles (sequence positions) can be determined as follows:

$$\Theta_i = \left\{t \in \{1,\ldots,T\} : \left|j - t \cdot r_{a(i)}\right| \leq DEV \wedge \left|j - 1 - (t-1) \cdot r_{a(i)}\right| \leq DEV\right\} \quad \forall\, i = 1,\ldots,T, \tag{9}$$

with $j = oc(\sigma', a(i), i)$ being the number of copies of model i's type $a(i)$ scheduled up to initial sequence position i. Thus, for a sequence position t being feasible for model i, it must hold that producing actual model i in period t does not surmount given deviation level DEV (first condition). Moreover, it must hold that postponing production of i to cycle t and, thus, having assigned only $j-1$ copies of model $a(i)$ up to the preceding cycle $t-1$, does not lead to excessive deviation in cycle $t-1$ (second condition).

With these sets on hand, the feasibility problem reduces to determining a perfect matching in a bipartite graph (e.g., see Hopcraft and Karp, 1973). Both node sets are determined by models $i = 1,\ldots,T$ and production cycles $t = 1,\ldots,T$, respectively, where i and t are connected by an arc whenever $t \in \Theta_i$ holds. A perfect matching denotes a feasible model sequence σ.

For restricting the number of feasibility problems to be solved, Steiner and Yeomans (1993) proved that merely the following set Γ of possible maximum deviation values have to be considered:

$$\Gamma = \left\{DEV \in \mathbb{R} : \min_{m \in M}\{1 - r_m\} \leq DEV \leq 1 \wedge DEV \cdot T \in \mathbb{Z}^+\right\} \tag{10}$$

A binary search within set Γ then delivers the minimum deviation, for which a feasible sequence can be determined, and the SY-procedure terminates.

The modifications required, so that SY can also solve PRVR, are twofold. First, when determining the sets Θ_i' of feasible production cycles per model i in addition to equation (9) a third condition must hold, so that limited re-sequencing flexibility is considered while solving feasibility problems:

$$\Theta'_i = \Theta_i \setminus \{t \in \{1,\dots,T\} \mid i - t > K\} \quad \forall i = 1,\dots,T, \tag{11}$$

Furthermore, the set Γ of possible maximum deviations is to be modified.

Let $\mu_t = \{a(i) \mid i \in \{1,\dots,T\} : t \in \Theta'_i\}$ be the set of model types that can be assigned to position t then Γ' is determined by:

$$\Gamma' = \left\{ DEV \in \mathbb{R} : \min_{m \in \mu_1} \{1 - r_m\} \leq DEV \leq DEV^{max} \wedge DEV \cdot T \in \mathbb{Z}^+ \right\} \tag{12}$$

The lower bound is strengthened by taking into account that not all model types can be assigned to the first position of the sequence. DEV^{max} can be set to the objective value of the initial sequence, which constitutes a simple upper bound as the unchanged sequence is feasible for PRVR, too.

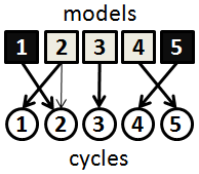

Figure 3: Perfect matching (bold faced) for $DEV = \frac{2}{5}$

Example (cont.): The initial sequence produces a maximum deviation $DEV^{max} = \frac{3}{5}$ caused by models 1 and 4. Since both model types are feasible at the first position ($\mu_1 = \{1, 2\}$) it follows that $DEV \geq min\left\{1 - \frac{3}{5}, 1 - \frac{2}{5}\right\}$, so that the set of possible deviations is $\Gamma' = \{\frac{2}{5}, \frac{3}{5}\}$. For a given deviation of $DEV = \frac{2}{5}$, we get $\Theta'_1 = \{2\}$, $\Theta'_2 = \{1, 2\}$, $\Theta'_3 = \{3\}$, $\Theta'_4 = \{5\}$, and $\Theta'_5 = \{4\}$. The resulting graph is depicted in Figure 3. As a perfect matching exists for $DEV = \frac{2}{5}$, the optimal reshuffled sequence is $\sigma = \{2, 1, 3, 5, 4\}$ with $Z^2(\sigma) = \frac{2}{5}$.

As DEV^{max} is set to the objective value of the initial sequence, the maximum run-time of the modified algorithm depends on the worst-case quality of any initial sequence. Let m^* be the model type with the highest demand, i.e., $d_{m^*} = \max\{d_m | m \in M\}$, then the worst-case (maximum) objective value can be easily determined by assigning all models of type m^* to the first (last) d_{m^*} positions of the sequence. It follows that $DEV^{max} \leq d_{m^*} \cdot (1 - r_{m^*}) < T$, however, since objective values can be fractional, the total number of values to be tested can be considerably higher. By equation (12) any element of set Γ' has to be a multiple of $1/T$, so that $|\Gamma'| < T^2$. Since checking feasibility for a given DEV can be done in $O(T)$ and the elements of Γ' can be efficiently generated and inspected using binary search, the total run-time of the modified algorithm is polynomial in T and of the order $O(T \log T)$.

2.4 Computational study

As exact polynomial time procedures have been presented for PRVR, computational performance of these procedures must not be reported in detail. Even with $T = 400$ cycles, which is a representative number of cars produced per shift in automobile industry, executing our PRVR procedures requires less than a second. Instead, we investigate the interdependency between improving solution quality of the initial sequence and the number of pull-off tables available.

The instances applied for this test are derived as follows. For a given number T of cycles and number $|M|$ of model types, first, each type receives a equally distributed random number out of interval $[0, 1]$. Then, the number of copies per model type is determined proportionally to the random numbers drawn, so that in total T models are to be produced. Finally, these copies are shuffled and, thus, randomly distributed over the initial sequence. The number T of cycles and the number $|M|$ of model types is varied as follows: (a) $|M| = 20$ and $T \in \{50, 100, 150, 200\}$ and (b) $T = 100$ and $|M| \in \{10, 20, 30, 40\}$. For each parameter constellation instance generation is repeated 100 times, so that 800 instances are derived. Each of these instances is solved for 22 different numbers of pull-off tables ($K \in \{0, 1, \ldots, 20, T - 1\}$) with either the sum-squared and the max-abs solutions procedure, so that in total 35,200 solutions are gained. All algorithms were implemented in C# 2008 and all tests have been conducted on a 2.1 GHz x86 Personal Computer with 2 GB of memory.

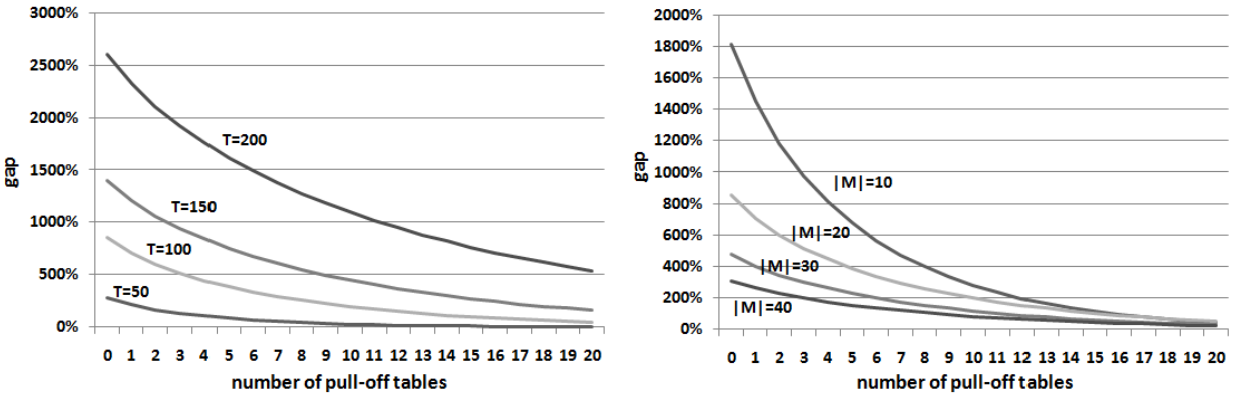

Figure 4: Average gap of PRVR with sum-squared-objective depending on number K of pull-off tables

Figure 4 summarizes the results for the sum-squared-objective and parameter constellations (a) and (b) on the left-hand and right-hand side, respectively. The performance indicator "gap" denominates the average relative deviation of the objective value gained with the current number of pull-off tables and the best possible solution value with full re-sequencing flexibility, i.e., the objective value of unrestricted PRV which is obtained by setting $K = T - 1$ in PRVR. Clearly, with increasing number of pull-off tables and, thus, increasing re-sequencing flexibility better (more leveled) solutions can be gained, with the incremental benefit of additional tables decreasing.

Our results deviate from those previously published by Lahmar et al. (2003) and Lahmar and Benjaafar (2007), who investigate re-sequencing in a paint-shop environment to batch cars of identical color and report that less than a hand-full tables is required to nearly reach full re-sequencing flexibility. In our setting, a considerable number of pull-off tables is required before getting sufficiently leveled sequences. For instance, with $T = 200$ cycles and $K = 20$ tables, the gap still amounts to a remarkable 534%. Thus, the number of tables to be installed heavily depends on the length of the sequence to be reshuffled. Note that, however, calculating the adequate number of pull-off tables in a real-world LS setting is a complex task since the benefit of a more or less leveled sequence can hardly be calculated accurately. The number of tables required also depends on the number of models to be produced (right-hand side of Figure 4). Counter to intuition, the gap widens with fewer models. However, with fixed number of cycles fewer models lead to a higher demand per model, which in turn raises target rates and, thus, penalizes deviations more severe.

Figure 5 depicts the results for the max-abs objective. As this case shows similar gap curves the aforementioned results hold, as well.

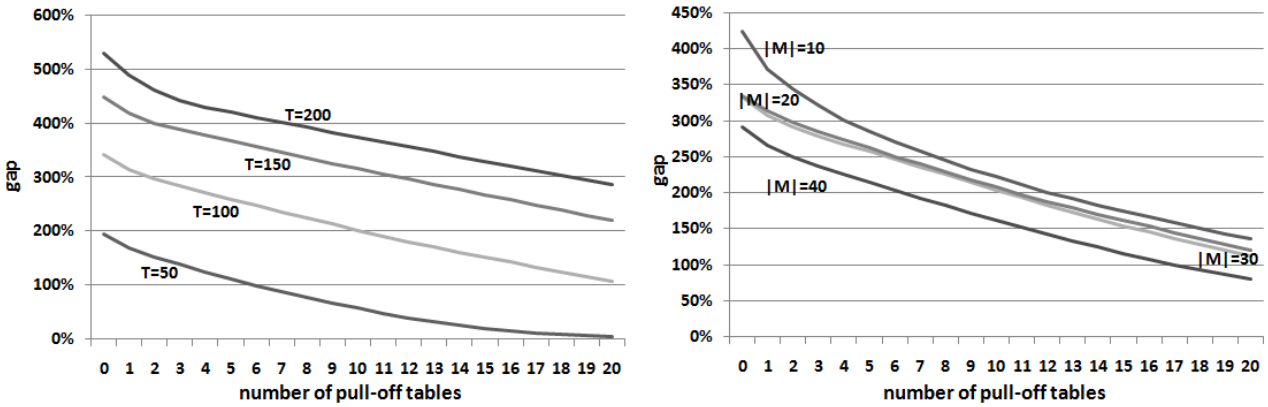

Figure 5: Average gap of PRVR with max-absolute-objective depending on number K of pull-off tables

3 The Output Rate Variation problem under limited re-sequencing flexibility (ORVR)

3.1 Problem description

The more detailed ORV explicitly considers the parts $p \in P$ required by each model type $m \in M$, expressed by part coefficients a_{pm}. Thus, the re-sequencing version of ORV (called ORVR) aims at a reshuffled model sequence σ, such that actual part demands $\delta(\sigma, p, t)$, defined as

$$\delta(\sigma, p, t) = \sum_{\tau=1}^{t} a_{p,a(\sigma^{-1}(\tau))} \quad \forall p \in P;\, t = 1, \ldots, T, \tag{13}$$

approximate target demands $t \cdot r_p$, where r_p denotes the target rate defined for each part p:

$$r_p = \frac{\sum_{i=1}^{T} a_{p,a(i)}}{T} = \frac{\sum_{m \in M} a_{pm}}{T} \quad \forall p \in P \tag{14}$$

Thus, within ORVR a mapping $\sigma : \{1, \ldots, T\} \rightarrow \{1, \ldots, T\}$ is to be determined, which minimizes objective function (15) subject to constraints (1), representing limited re-sequencing flexibility:

$$\text{minimize } Z^3(\sigma) = G\left(F_p\left(\delta(\sigma, p, t) - t \cdot r_p\right)\right). \tag{15}$$

Note that again $G(\cdot)$ and $F_p(\cdot)$ denote different forms of aggregation functions, i.e., the sum- and max-function, and (possibly part specific) deviation functions, i.e., absolute and squared deviations, respectively. Further note that the ORVR with a facultative number of pull-off tables is NP-hard in the strong sense. This is obviously true because with $K \geq T - 1$, ORVR is not restricted in its assignment decision (of models to cycles) and the traditional ORV arises. The ORV with different aggregation and deviation functions was shown to be NP-hard in the strong sense (Kubiak, 1993 and Kubiak et al., 1997).

In the following we will discuss a general graph approach, which can simply be adopted for all aforementioned aggregation and deviation functions. We build up on the research of Lim and Xu (2009), who propose a dynamic programming approach to solve a re-sequencing problem with pull-off tables which batches cars to blocks of identical color in front of the paint shop of a mixed-model assembly line. We show how to adapt and improve their approach for an application to LS by using the concepts proposed by Kubiak et al. (1997). As customizing the graph search for different functions is readily available, we will restrict our description to the sum-squared-case.

3.2 A graph search procedure

The graph approach is based on an acyclic digraph $G(V, E, r)$ with a node set V divided into $T \cdot (K+1)+1$ stages, a set E of arcs connecting nodes and an arc weighting function $r : E \rightarrow \mathbb{R}$. To define node set V, it is necessary to examine possible decisions at a sequence position i. Lim and Xu (2009) differentiate three types of decisions, which can be taken for each model i of the initial sequence:

- Move current model i into a pull-off table, if an empty table exists.
- Produce current model i, while leaving models in pull-off table unchanged.
- Reinsert and produce a model from a pull-off table (if at least one table contains any model).

As the alternative decisions at a current decision point only depend on model i being the next in line and the current content (set of stored models) κ of pull-off tables and, furthermore, the impact of these alternatives on the objective value only depends on the models (and their part requirements) previously scheduled, a node representing a current decision point can be defined as $[i, \kappa]$.

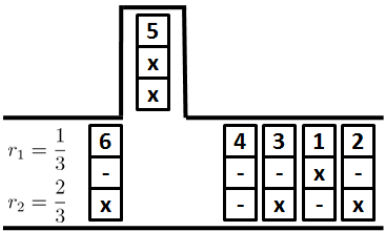

Figure 6: Example for a decision point in ORVR with six models and a single pull-off table

Example: Consider the example depicted in Figure 6. Here, a current decision point ($t = 5$) of the re-sequencing process is depicted, where four models are already fixed while models 5 and 6 wait off-line and on-line, respectively. While model 5 requires both parts ($a_{15} = a_{25} = 1$ indicated by "x"), model 6 requires $a_{16} = 0$ and $a_{26} = 1$ units of part 1 and 2, respectively (indicated by "-" and "x"). Furthermore, the resulting target rates $r_1 = \frac{2}{6} = \frac{1}{3}$ and $r_2 = \frac{4}{6} = \frac{2}{3}$ for part $p = 1$ and $p = 2$, respectively, are given in Figure 6. The node for the current state is defined as follows: $[6, \{5\}]$. When either model 5 or model 6 is produced, successor nodes $[6, \emptyset]$ and $[7, \{5\}]$ are to be branched, respectively.

Node set V is subdivided into $T \cdot (K + 1) + 1$ stages, where a stage (j, k) contains all nodes $V_{(j,k)} \subset V$, where j models are definitely fixed to the first j positions of the sequence and $k = |\kappa|$ models are stored in pull-off tables. This way, a forwardly directed graph arises, which means that an arc can only point from a node of stage (j, k) to a node of stage (j', k'), if $j < j' \vee (j = j' \wedge k < k')$ holds. In particular, a node of stage (j, k) can only be connected with nodes of the following stages: $(j, k+1)$ (put current model in pull-off table), $(j+1, k-1)$ (reinsert model from pull-off) or $(j + 1, k)$ (produce current model). This way, a stage-wise generation of the graph and a simultaneous evaluation of the shortest path to any node is enabled. Obviously, first stage $(0, 0)$ and final stage $(T, 0)$ contain merely a single node $[1, \emptyset]$ (start node) and $[T+1, \emptyset]$ (sink node), respectively.

Arcs of arc set E connect nodes of adjacent stages and thus represent a transition between two decision points. The result of such a transition is a decision on the current model, so that a value $j > 0$ representing the current model fixed in the final sequence is stored with each arc. On the other hand, $j = 0$ is to be stored with the arc, if the current model is moved into pull-off table and no model is definitely fixed. There exist the aforementioned three kinds of transitions from any current node $[i, \kappa]$, so that the following cases are to be distinguished:

- If $|\kappa| < K$, there exists at least one empty pull-off table, so that current model i can be pulled into off-line buffer and node $[i+1, \kappa \cup \{i\}]$ is to be generated. Thus, no model is fixed in the final sequence (produced) and $j = 0$ is stored with the arc. For example, consider the preceding decision point of that depicted in Figure 6 before pulling model 5 off-line. Here, the node $[5, \emptyset]$ is developed to depicted node $[6, \{5\}]$ and both nodes are connected by an arc with $j = 0$.

- Furthermore, current on-line model i can directly be produced while pull-off tables remain unchanged. We can, however, make use of the following observation here: Since models of the same type have identical part demands by definition, they can always swap positions in a sequence without changing cumulated deviations. It follows, that if models of the same type would overtake each other in the reshuffled sequence, then we could simply swap these two models and restore *order preservation*. As a consequence, we do not need to branch a node for an on-line model i, if a model of the same type was stored in the off-line buffer. Therefore, an arc is to be inserted pointing to node $[i+1, \kappa]$, only if $\nexists j \in \kappa | a(i) = a(j)$. In that case, model i is stored with the arc as the current model produced. Consider Figure 6 as the present decision point and on-line model 6 to be produced. Here, an arc (with which model $j = 6$ is stored) is to be inserted from current node $[6, \{5\}]$ to successor node $[7, \{5\}]$, since no model of the same type exists in buffer.

- Finally, if $\kappa \neq \emptyset$, any off-line model waiting in the pull-off table can be reinserted and produced. Due to the described order preservation, it is sufficient to branch a successor node for the model with the smallest index number only, whenever more than one model of the same type is stored in the buffer. Therefore for each $j \in \kappa$ for which $\nexists j' \in \kappa | a(j) = a(j') \wedge j' < j$ a successor node $[i, \kappa \setminus \{j\}]$ is to be generated and connected by an arc with j being the current model produced and stored with the arc. In Figure 6 model 5 can be reinserted from the pull-off table, so that an arc (with which model $j = 5$ is stored) is to be generated connecting current node $[6, \{5\}]$ and successor node $[6, \emptyset]$.

As it is not necessary to generate duplicate nodes within a stage, in a computer implementation of the graph, nodes per stage can be stored efficiently in a hash-table and addressed by a unique hash-key.

Finally, arc weights $c : E \rightarrow \mathbb{R}$ are assigned to each arc, which store the contribution of the current sequencing decision to the overall objective value. Note that this contribution cannot be stored with nodes, because then moving a model into pull-off table would cause an additional deviation. Thus, an arc weight becomes zero whenever an arc represents the transition of pulling a model into pull-off table. If a model is definitely fixed at the current decision point, a weight $c_{([i',\kappa'],[i,\kappa])}$ belonging to arc $([i', \kappa'], [i, \kappa]) \in E$ is calculated as follows:

$$c_{([i',\kappa'],[i,\kappa])} = \sum_{p \in P} \left(\sum_{j \in \{1,\ldots,i-1\} \setminus \kappa} a_{pa(j)} - (i - 1 - |\kappa|) \cdot r_p \right)^2 \quad \forall\, ([i',\kappa'],[i,\kappa]) \in E \tag{16}$$

Here, the resulting set of models already fixed can be easily determined from the information stored with each destination node $[i,\kappa]$, which are all those models with $1 \leq j \leq i-1$ minus those still being off-line (stored in κ). These fixed models determine the actual demand per part p, which is to be compared with target demand calculated by multiplying the current number of production cycles $t = i - 1 - |\kappa|$ already fixed with target demand rate r_p. Note that right here deviation function is to be exchanged, if absolute deviations are to be considered instead.

Example (cont.): In our example of Figure 6, current node $[6,\{5\}]$ is to be branched into nodes $[6,\emptyset]$ (reinsert off-line model 5) and $[7,\{5\}]$ (produce on-line model 6), respectively. The former choice leads to a squared deviation of $c_{([6,\{5\}],[6,\emptyset])} = 0.\overline{5}$ and the latter to $c_{([6,\{5\}],[7,\{5\}])} = 2.\overline{8}$.

With this graph on hand solving ORVP reduces to finding the shortest path from start node $[1,\emptyset]$ to sink node $[T+1,\emptyset]$, which, following the principle of dynamic programming, must not separately be calculated after constructing the complete graph but can simultaneously derived by stage-wise storing the shortest path to each node. For our example of Figure 6 the optimal reshuffled sequence is $\sigma = \{2,1,3,4,5,6\}$ with objective value $Z^3(\sigma) = 1.\overline{2}$. Note that, instead, the min-max path is to be stored with each node, if, e.g., the max-abs case of ORVR is considered.

As the size of the graph increases exponentially with the number K of pull-off tables, exploring the complete graph by exhaustive search will be too time-consuming for larger instances. Instead, a heuristic graph search seems better suited. Well known meta-heuristic Beam Search (BS) (e.g., see Lowerre, 1976; Ow and Morton, 1988) is such a graph search procedure, which heuristically restricts the set of nodes per stage to be further branched to a promising subset. This choice is typically being based on a priority value (see Sabuncuoglu et al., 2008) and in its most basic form this value is simply the partial objective value of the shortest path to the respective node. With these priority values on hand, BS chooses the BW best nodes per stage to be further branched while excluding the rest, with beam width BW being the basic control parameter. With our re-sequencing graph and shortest-path lengths as priority values, a BS procedure for ORVR is readily available.

3.3 Computational study

The computational part for ORVR aims at investigating the solution performance of our exact (exhaustive search) and heuristic (Beam Search - BS) procedures. Furthermore, the impact of an increasing number of pull-off tables on solution quality is tested. To answer the former research question 70 test instances are derived by randomly generating 10 instances per varying number

of cycles: $T \in \{10, 15, \ldots, 40\}$. Each instance is derived by randomly setting each product coefficient of the bill of material ($T \cdot |P|$-matrix) to one with a given probability of $Prob = 0.5$. These instances are solved for $K \in \{0, 1, \ldots, 6\}$ pull-off tables and with six different solution procedures (exhaustive search + BS with five different beam widths $BW \in \{2, 5, 20, 50, 100\}$), so that in total 2,940 solution runs have been executed.

First, Figure 7 depicts the solution time (in CPU-seconds) of exact exhaustive search for different number of cycles and pull-off tables. The results confirm a linear increase in the number T of cycles but an exponential increase in the number K of pull-off tables. With $K = 6$ and $T = 40$, average solution time amounts to 67.4 CPU-seconds, so that the upper limit, up to which exhaustive search can reasonably be applied, ranges near these parameter values.

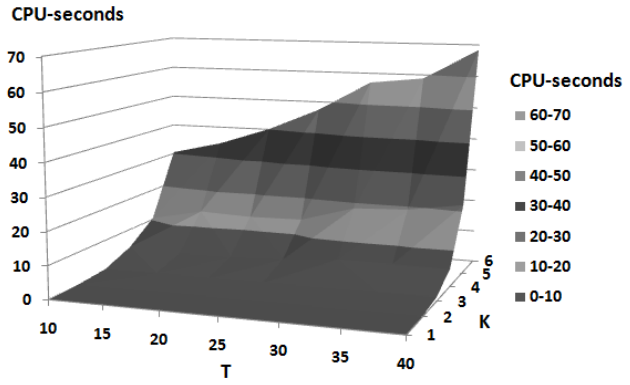

Figure 7: Performance (in CPU-seconds) of exhaustive search depending on sequence length T and number K of pull-off tables

The solution quality of heuristic BS with diverging beam widths BW is reported in Table 1 by listing the average CPU-seconds (cpu) and the average relative gap (gap in %) between heuristic and optimal solution values. The results reveal a very good solution quality within a very short time frame. For instance, with a beam width of $BW = 100$ BS solves 96% of all instances to optimality in merely 0.24 CPU-seconds on average. Thus, beam search seems well suited for solving large ORVR instances and we restrict the investigation of research question two, the impact of a varying number of pull-off tables, to heuristic solutions gained with BS ($BW = 20$).

For this purpose, random bills of material are derived in the aforementioned manner for diverging numbers of cycles: $T \in \{50, 100, 150, 200\}$. Instance generation is repeated 100 times, so that 400 instances are derived. These instances are solved with BS for 20 different numbers of pull-off tables ($K \in \{1, \ldots, 20\}$), so that in total 8,000 solution values are determined. The results are depicted in Figure 8. Here, the average relative improvement (imp in %) of the solution gained with the respective number of pull-off tables in relation to the solution value of the initial sequence (with $K = 0$) is reported. Obviously, only a few pull-off tables are required to reduce the total deviation of the initial sequence dramatically as about 6 (for $T = 50$) to 12 (for $T = 200$) tables are sufficient to reduce the deviations to about 5% of the initial value. As in

		$BW = 2$		$BW = 5$		$BW = 20$		$BW = 50$		$BW = 100$	
K	T	gap	cpu	gap	cpu	gap	cpu	gap	cpu	gap	cpu
1	10	3%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	15	3%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	20	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	25	2%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	30	1%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	35	2%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	40	3%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
2	10	7%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	15	8%	<0.1	1%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	20	4%	<0.1	2%	<0.1	1%	<0.1	0%	<0.1	0%	<0.1
	25	8%	<0.1	2%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	30	5%	<0.1	1%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	35	9%	<0.1	3%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
	40	9%	<0.1	6%	<0.1	0%	<0.1	0%	<0.1	0%	<0.1
3	10	17%	<0.1	6%	<0.1	0%	<0.1	0%	0.1	0%	0.1
	15	12%	<0.1	2%	<0.1	0%	<0.1	0%	0.1	0%	0.1
	20	10%	<0.1	5%	<0.1	0%	<0.1	0%	0.1	0%	0.1
	25	15%	<0.1	1%	<0.1	0%	<0.1	0%	0.1	0%	0.1
	30	15%	<0.1	3%	<0.1	0%	<0.1	0%	0.1	0%	0.1
	35	12%	<0.1	3%	<0.1	1%	<0.1	1%	0.1	0%	0.1
	40	14%	<0.1	5%	<0.1	0%	<0.1	0%	0.1	0%	0.1
4	10	25%	<0.1	8%	<0.1	1%	0.1	0%	0.1	0%	0.2
	15	17%	<0.1	7%	<0.1	2%	0.1	0%	0.1	0%	0.2
	20	15%	<0.1	5%	<0.1	0%	0.1	0%	0.1	0%	0.2
	25	22%	<0.1	3%	<0.1	0%	0.1	0%	0.1	0%	0.2
	30	22%	<0.1	5%	<0.1	0%	0.1	0%	0.1	0%	0.2
	35	15%	<0.1	6%	<0.1	3%	0.1	2%	0.1	0%	0.2
	40	8%	<0.1	4%	<0.1	1%	0.1	0%	0.1	0%	0.2
5	10	28%	<0.1	7%	<0.1	2%	0.1	1%	0.2	0%	0.4
	15	24%	<0.1	10%	<0.1	1%	0.1	1%	0.2	1%	0.4
	20	11%	<0.1	5%	<0.1	2%	0.1	2%	0.2	0%	0.4
	25	28%	<0.1	10%	<0.1	0%	0.1	0%	0.2	0%	0.4
	30	22%	<0.1	7%	<0.1	2%	0.1	1%	0.2	0%	0.4
	35	21%	<0.1	11%	<0.1	2%	0.1	0%	0.2	0%	0.4
	40	21%	<0.1	9%	<0.1	2%	0.1	1%	0.2	0%	0.4
6	10	24%	<0.1	9%	<0.1	1%	0.1	1%	0.4	0%	0.7
	15	18%	<0.1	7%	<0.1	2%	0.1	1%	0.3	1%	0.7
	20	20%	<0.1	9%	<0.1	3%	0.1	1%	0.3	1%	0.7
	25	22%	<0.1	7%	<0.1	1%	0.1	0%	0.3	0%	0.6
	30	20%	<0.1	5%	<0.1	1%	0.1	0%	0.3	0%	0.6
	35	24%	<0.1	11%	<0.1	5%	0.1	1%	0.3	1%	0.6
	40	17%	<0.1	7%	<0.1	2%	0.1	0%	0.3	0%	0.6
total		14%	<0.1	5%	<0.1	1%	0.1	0%	0.1	0%	0.2

Table 1: Solution quality of Beam Search with diverging beam widths (BW)

case of PRVR, the results show that an increasing number T of cycles increases the number of pull-off tables required to level material demand.

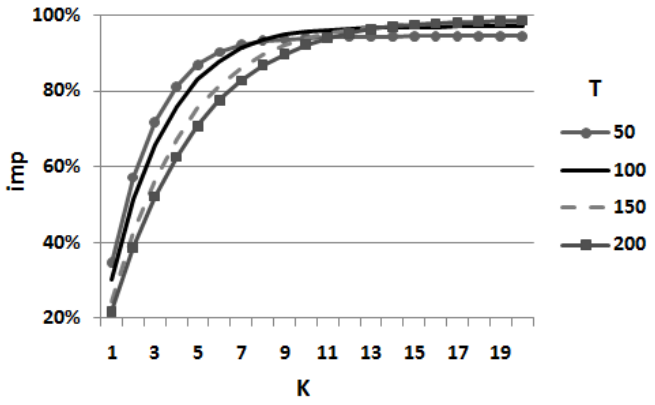

Figure 8: Average improvement (imp) of initial solution for ORVR with sum-squared-objective depending on the number K of pull-off tables

4 On the ability of PRVR approximating ORVR

In this section, it is investigated whether (or under which prerequisites) PRVR is a suited approximation of ORVR. Recall that such a replaceability is desirable since PRVR can be solved to optimality in polynomial time, whereas ORVR is NP-hard in the strong sense. With regard to traditional level scheduling, literature claims such a replaceability between both models whenever:

- *Miltenburg-case:* "Products require approximately the same number and mix of parts." (Miltenburg 1989, p. 193).
- *Kubiak-case:* "Outputs [of preceding production levels] required for each different product are distinct." (Kubiak 1993, p. 261).

In a recent paper, Boysen et al. (2009b) question these statements by determining PRV solutions, evaluating them with the ORV objective function, and comparing these results with model sequences directly gained by an ORV procedure. The results reveal enormous deviations of PRV solutions whenever the aforementioned premises do not hold and parts occur in varying composition in their respective models. Furthermore, it is argued that both premises are seldom given in the real-world, because typically customers define products according to their individual needs (mass-customization). Thus, it is concluded that more aggregate PRV is indeed not applicable in today's mixed-model assembly lines to reasonably approximate more detailed ORV. This section transfers the computational tests of Boysen et al. (2009b) to the re-sequencing version of LS.

For this purpose, test instances are derived by systematically varying parameter *Prob*, which defines the probability of each product coefficient of the bill of material of either requiring the

respective part ($a_{pm} = 1$) or not ($a_{pm} = 0$) (see Boysen et al, 2009b). With $Prob$ being close to zero sparse matrices result. Thus, models require few and (most probably) divergent parts, so that instances representing the Kubiak-case arise. On the other hand, dense bill of material-matrices obviously approach the Miltenburg-case with models requiring many parts in similar composition. Consequently, we systematically vary $Prob \in \{i \cdot 0.05 \mid i = 1, \ldots, 19\}$, so that a continuum of part commonality between both extremes (the Kubiak- and Miltenburg-case) arises. Specifically, for a given probability $Prob$, number T of cycles, number $|P|$ of parts and number K of pull-off tables, each instance is derived as follows: First, an enlarged bill of material ($T \cdot |P|$-matrix) is randomly generated according to given probability $Prob$. Then, this matrix is condensed by joining equal columns (model copies), so that the final bill of material of size $|M| \cdot |P|$ results. For each of two chosen parameter constellations ($T = 100; |P| = 5; K = 5$ and $T = 100; |P| = 10; K = 10$) instance generation is repeated 100 times, so that $2 \cdot 19 \cdot 100 = 3,800$ instances result. Any instance is solved by PRVR and ORVR procedures with both max-abs and sum-squared objective function, so that in total 15,200 solution runs are executed. For solving each PRVR instance, the respective exact solution procedure presented in Section 2 is chosen. The resulting sequences are then evaluated with the respective ORV objective function and these results are compared to heuristic ORVR solutions directly determined by BS (with $BW = 20$) as described in Section 3. The results of this comparison are summarized in Figure 9.

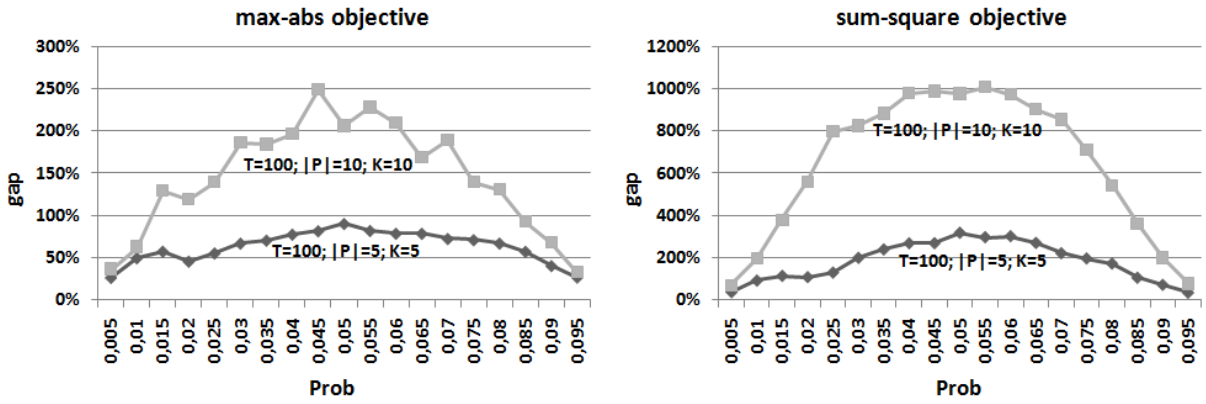

Figure 9: Gap between PRVR (evaluated with ORV objective function) and ORVR for diverging probabilities $Prob$

Figure 9 reveals tremendous average gaps between PRVR and ORVR up to 1,008%, i.e., the PRVR solutions have much larger ORV objective values than the heuristic ORVR solutions. These gaps widen if (i) the sum-squared objective is applied, (ii) instances increase in size and (iii) models increasingly share parts in diverging composition. Obviously, already slight deviations from the extremes (Kubiak- and Miltenburg-case) lead to considerable differences between both approaches. Thus, it can be concluded that (in analogy to traditional LS as documented by Boysen et al., 2009b) PRVR is not a suited approximation for ORVR and it is much more promising to directly solve ORVR even if only heuristic solutions can be determined.

5 Conclusion

The paper on hand investigates the level scheduling problem, which aims at an even distribution of material demands over time, in a re-sequencing environment. A given number of pull-off tables is available to reshuffle a given initial sequence, which was either changed by unforseen events like defects or initially planned to serve the objective of a preceding production stage. For this purpose, the traditional forms of level scheduling, namely the Product Rate Variation (PRV) and the Output Rate Variation (ORV) problem, are adopted to the re-sequencing environment and suited solution procedures are introduced and tested in comprehensive computational studies. As a main result, it is shown that with an increasing number of production cycles to be leveled the number of pull-off tables required increases considerably. Furthermore, it is shown that the PRV is no suited approximation of the ORV in our re-sequencing environment.

Future research could deal with more efficient solution procedures for ORVR. Furthermore, the re-sequencing versions of other well known approaches for sequencing mixed-model assembly lines, i.e., car-sequencing and mixed-model sequencing, have not yet been considered. Thus, specifying the respective problems and developing efficient solution procedures would be a valuable contribution to further streamline real-world mixed-model assembly systems.

References

[1] Boysen, N., Fliedner, M., Scholl, A., 2009a. Sequencing mixed-model assembly lines: Survey, Classification and Model Critique, European Journal of Operational Research 192, 349–373.

[2] Boysen, N., Fliedner, M., Scholl, A., 2009b. The product rate variation problem and its relevance in real world mixed-model assembly lines. European Journal of Operational Research 197, 818–824.

[3] Dhamala, T.N., Kubiak, W., 2005. A brief survey of just-in-time sequencing for mixed-model systems. International Journal of Operations Research 2, 38–47.

[4] Ding, F.-Y., Sun, H., 2004. Sequence alteration and restoration related to sequenced parts delivery on an automobile mixed-model assembly line with multiple departments. International Journal of Production Research 42, 1525–1543.

[5] Gusikhin, O., Caprihan, R., Stecke, K.E., 2008. Least in-sequence probability heuristic for mixed-volume production lines. International Journal of Production Research 46, 647–673.

[6] Hopcroft, J., Karp, R., 1973. An n2.5 algorithm for maximum matching in bipartite graphs. SIAM Journal on Computing 2, 225–231.

[7] Inman, R.R., Schmeling, D.M., 2003. Algorithm for agile assembly-to-order in the automotive industry. International Journal of Production Research 41, 3831–3848.

[8] Kubiak, W., 1993. Minimizing variation of production rates in just-in-time systems: A survey. European Journal of Operational Research 66, 259–271.

[9] Kubiak, W., Sethi, S.P., 1991. A note on "level schedules for mixed-model assembly lines in just-in-time production systems". Management Science 37, 121–122.

[10] Kubiak, W., Sethi, S.P., 1994. Optimal just-in-time schedules for flexible transfer lines. The International Journal of Flexible Manufacturing Systems 6, 137–154.

[11] Kubiak, W., Steiner, G., Yeomans, J.S., 1997. Optimal level schedules for mixed-model, multi-level just-in-time assenbly systems. Annals of Operations Research 69, 241–259.

[12] Lahmar, M., Benjaafar, S., 2007. Sequencing with limited flexibility. IIE Transactions 39, 937–955.

[13] Lahmar, M., Ergan, H., Benjaafar, S., 2003. Resequencing and feature assignment on an automated assembly line. IEEE Transactions on Robotics and Automation 19, 89–102.

[14] Lim, A., Xu, Z., 2009. Searching optimal resequencing and feature assignment on an automated assembly line. Journal of the Operational Research Society 60, 361–371.

[15] Lowerre, B.T., 1976. The HARPY speech recognition system. Ph.D. thesis, Carnegie-Mellon University, U.S.A., April.

[16] Miltenburg, J., 1989. Level Schedules for mixed-model assembly lines in just-in-time production systems. Management Science 35, 192–207.

[17] Monden, Y., 1998. Toyota Production System: an integrated approach to just-in-time. 3rd edition, Norcross, 1998.

[18] Ow, P.S., Morton, T.E., 1988. Filtered beam search in scheduling. International Journal of Production Research 26, 297–307.

[19] Sabuncuoglu, I., Gocgun, Y., Erel, E., 2008. Backtracking and exchange of information: Methods to enhance a beam search algorithm for assembly line scheduling. European Journal of Operational Research 186, 915–930.

[20] Spieckermann, S., Gutenschwager, K., Voss. S., 2004. A sequential ordering problem in automotive paint shops. International Journal of Production Research 42, 1865–1878.

[21] Steiner, G., Yeomans, J.S., 1993. Level schedules for mixed-model, just-in-time processes. Management Science 39, 728–735.