Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A.A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

APPLICATION OF ONLINE LEARNING FOR THE DYNAMIC CONFIGURATION OF KANBAN SYSTEMS

Daniel Rippel
Michael Lütjen

BIBA – Bremer Institut für Produktion und Logistik GmbH at the University of Bremen
Hochschulring 20
Bremen, 28359, GERMANY

Michael Theß

Signal Cruncher GmbH

Franz-Jacob-Str. 2A
Berlin, 10369, GERMANY

Michael Freitag

University of Bremen, Faculty of Production Engineering
Bibliothekstraße 1
Bremen, 28359, GERMANY

ABSTRACT

Kanban systems constitute a widely used pull control for inventory management in production systems. As a result of an increasingly volatile and individualized customer demand, Kanban systems have to be reconfigured dynamically to achieve minimal inventory levels while maintaining a stable production. This paper investigates the application of an incremental online learning platform called XELOPRO to optimize inventory levels using the current state of the production system, while including contextual information, e.g., time-related information. As the platform uses an incremental support vector machine to update its models during runtime without the need to store and reevaluate large amounts of historical data, it constitutes a suitable tool for a decentralized inventory management. Results show a good performance with drastic decreases in inventory levels compared to static configurations and a higher reliability compared to a dynamic application of standard Kanban rules.

1 INTRODUCTION

Over the last decades, just-in-time production concepts became increasingly important in several industrial branches (Majava and Ojanperä 2017). One of the major concepts includes the inventory management, i.e., the availability of required materials at production or assembly stations at the right time in the right quantity. A widely applied concept for inventory management is the Kanban control system. The Kanban system was originally developed by Toyota in the 1970s and has since been extended and improved in theory and practice (Zhang et al. 2015). The system constitutes a decentralized pull control for production or inventory management, whereby it basically consists of a number of cards, each representing one batch of material (Ohno 2013). Thereby, each card is attached to a container that contains a specified amount of material. When the container is empty, the Kanban card is removed and posted on a Kanban board, triggering a resupply order for a new container of this material. The configuration of such a Kanban system usually consists of either determining the required amount of Kanban, i.e., of cards and, therefore, the number of containers within the system, or of determining the required amount of material per container, depending on which of both numbers is considered to be fixed. The overall objective of such systems is the

minimization of work-in-progress levels for material and products, while rendering demand fluctuations more controllable by decentralizing the production and inventory control (Kimura and Terada 2007).

As the Kanban control system was originally developed for the automotive industry, assembly lines still constitute a major example for their application. In case of inventory control, the assembly stations are assumed to include a local storage of material used for the assembly. Each storage contains a number of containers for the material, whereby restocking of this material is performed decentralized, using a Kanban control. In cases of a variant–rich production, each local storage contains several types of material, applicable to one or more product variants and to several products. As a result, the estimation of each part's required amount becomes increasingly difficult. If the demand for variants varies over time, e.g., periodically or even unpredictably by customer demand, the configuration of the respective Kanban systems has to be adapted dynamically to avoid a lack of materials or excessively high inventory levels.

This paper proposes the application of online learning techniques to the configuration of a Kanban inventory control for a variant-rich assembly scenario. The next section discusses several methods for dynamic Kanban control systems. Afterwards, the paper introduces the proposed online learning and optimization method, which is capable of adjusting the underlying regression models during runtime and uses them to optimize the Kanban system. Section 4 presents a simulation model and the results of a simulation study conducted to evaluate the performance of the online learning approach.

2 CONFIGURATION OF KANBAN SYSTEMS

For relatively static production systems, the Kanban control can be configured using a variety of mathematical equations or methods, generally dependent on the average amount of required material types, i.e., production times, demand, set-up times, etc., and on the time it takes to restock the inventory, i.e., to obtain a new container (cf. Shahabudeen et al. 2003, Belisário and Pierreval 2015). However, customization and volatile demand lead to high variations and dynamics in production systems, which require continuous adjustments of the Kanban systems.

In general, approaches for dynamic Kanban System adjustments usually focus on the estimation of the future required amounts of material and use these estimations to adjust the number of Kanbans within the system. One of the earliest methods goes back to Rees et al. (1987), who combined standard demand forecasts with a statistical analysis of historical lead times to estimate the required number of Kanban cards for the next period of time. Accordingly, Gupta and Al-Turki (2010) describe a flexible Kanban system, which differentiates between permanent and additional Kanban cards, whereby the additional cards can be added or removed based on fixed periods and expected variations in the customer demand. This approach assumes that the customer demand is known for the given period. More recent research relies on heuristics and simulation-based optimization to determine possible material shortages using the previously known production schedule (e.g., Talibi et al. 2013). Besides these approaches, which use predictions or estimations to determine the number of Kanban cards for the next period, another set of methods focusses on monitoring the current state of the system while adding or removing cards accordingly. Tardif and Maaseidvaag (2001) proposed an approach that also relies on permanent and additional cards. The approach monitors the relationship between demand and finished products to add additional cards dynamically if the relation shifts towards an increasing demand or removes cards if the demand decreases. This approach has been adapted later on for multi-stage Kanban systems (Sivakumar and Shahabudeen 2008; Takahashi et al. 2010) or for other key values, e.g., to achieve a desired throughput (Framinan et al. 2006).

Apart from methods directly adjusting the Kanban system, other approaches aim to influence the overall production system to ensure a stable production while not changing the Kanban configuration. For example, Korugan and Gupta (2014) add additional degrees of freedom by adding or removing workstations to the production system. Other approaches aim at altering the restock times by prioritizing Kanbans if the inventory level becomes increasingly low (Parente et al. 2012).

The methods described above either rely on reliable predictions of the future demand or focus on reacting to changes in the system. Especially for highly dynamic production systems, these predictions can be hard to acquire. On the other hand, purely reactive strategies neglect the additional value of contextual

information. To include such information without statically describing the behavior of the overall production system, this paper proposes to use online learning techniques to learn and refine a model of the systems characteristics and further use this model to configure the underlying Kanban system.

3 ONLINE LEARNING FOR A KANBAN CONFIGURATION

In order to optimize Kanban systems under dynamic influences, this paper investigates the application of the XELOPRO online learning platform, developed by the prudsys AG and Signal Cruncher GmbH (Signal Cruncher GmbH 2018). XELOPRO applies a two-stage learning algorithm, whereby the first stage uses offline learning to determine an initial model of the learnt context, while the second stage uses online learning to adapt the initially created model in real-time during application. As a baseline, the XELOPRO applies regression techniques for learning and supports a variety of methods, e.g., regression trees, linear and polynomial regression, spline regression, or advanced methods like Bayesian regression, factorization machines, (deep) neural networks, sparse grids, or support vector machines (SVM). In contrast to classical approaches, the supported methods are extended to enable an incremental refinement of the models, without the need to store and reevaluate the historical database.

In the current work, an SVM based on random features was selected which applies a linear formulation (Rahimi and Recht 2008). The formulation is based on the idea to approximate the SVM kernel $k(\mathbf{x}, \mathbf{y})$ with $\mathbf{x}, \mathbf{y} \in \mathfrak{R}^d$ using a randomized feature map $\mathbf{z}$: $\mathfrak{R}^d \rightarrow \mathfrak{R}^D$:

$$k(\mathbf{x}, \mathbf{y}) \approx \mathbf{z}(\mathbf{x})^{\mathrm{T}} \mathbf{z}(\mathbf{y}).$$

Here, d is the dimension of the original attribute space and D is a small dimension, usually not higher than several hundred. This approach does not only speed up the learning extremely, but it also provides a way to quickly evaluate the function. Indeed, instead of computing $f(\mathbf{x}) = \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x})$ at the test point $\mathbf{x}$, which requires $O(Nd)$ operations (N is the number of all training data points and c_i are the basis coefficients), the new formulation leads to $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \mathbf{z}(\mathbf{x})$ with $\mathbf{w}^{\mathrm{T}} = (w_1, \ldots, w_D)$ basis coefficients resulting in $O(D + d)$ operations only.

Moreover, the linear feature map gives rise to a highly efficient incremental learning procedure. To determine the vector of basis coefficients $\mathbf{w}$ we solve a regularized least-square problem

$$\min_{\mathbf{w}} R(\mathbf{w}) = \min_{\mathbf{w}} \left(\left\| \mathbf{Z}^{\mathrm{T}} \mathbf{w} - \mathbf{y} \right\|_2^2 + \lambda \|\mathbf{w}\|_2^2 \right) \tag{1}$$

where $\mathbf{y}$ denotes the vector of the target attribute values, $\mathbf{Z} = (\mathbf{z}(\mathbf{x}_1) \ \ldots \ \mathbf{z}(\mathbf{x}_N))$ the matrix of random features over the training vectors $\mathbf{x}_i$, and $\lambda > 0$ the regularization parameter. Instead of solving problem (1) directly, e.g., by a stochastic gradient method, we follow the guidelines of Paprotny and Theß (2013) in Chapter 7. For solving (1), we set

$$\frac{\sigma R(\mathbf{w})}{\sigma w_i} = 0, \ i = 1, \ldots, D$$

and after some transformations arrive at

$$\left(\mathbf{Z}\mathbf{Z}^{\mathrm{T}} + \lambda I \right) \mathbf{w} = \mathbf{Z}\mathbf{y} \tag{2}$$

Compared to the original formulation (1) in the large space of N dimensions, equation system (2) operates in the space of just D dimensions, which is usually much smaller. Further, the system (2) is symmetric positive definite and hence can be solved quickly, e.g., by a conjugate gradient method (CG). Even more, system (2) is well-suited for iterative updates. For M new training data points $\{(\hat{\mathbf{x}}_i, \hat{y}_i)\}_{i=1}^{M}$, system (2) becomes

$$\left(\bar{\mathbf{Z}}\bar{\mathbf{Z}}^{\mathrm{T}} + \lambda I\right)\bar{\mathbf{w}} = \bar{\mathbf{Z}}\bar{\mathbf{y}} \tag{3}$$

with the extended vector $\bar{\mathbf{y}} = (\mathbf{y}\ \hat{\mathbf{y}})^{\mathrm{T}} = (\mathrm{y}_1, \ldots, \mathrm{y}_N, \hat{\mathrm{y}}_1, \ldots, \hat{\mathrm{y}}_M)^{\mathrm{T}}$ and the extended matrix of random features $\bar{\mathbf{Z}} = \left(\mathbf{Z}\ \hat{\mathbf{Z}}\right) = (\mathbf{z}(\mathbf{x}_1)\ \ldots\ \mathbf{z}(\mathbf{x}_N)\ \mathbf{z}(\hat{\mathbf{x}}_1)\ \ldots\ \mathbf{z}(\hat{\mathbf{x}}_M))$. The update formulas

$$\bar{\mathbf{Z}}\bar{\mathbf{Z}}^{\mathrm{T}} = \left(\mathbf{Z}\ \hat{\mathbf{Z}}\right)\begin{pmatrix}\mathbf{Z}^{\mathrm{T}}\\ \hat{\mathbf{Z}}^{\mathbf{T}}\end{pmatrix} = \mathbf{Z}\mathbf{Z}^{\mathrm{T}} + \hat{\mathbf{Z}}\hat{\mathbf{Z}}^{\mathbf{T}},\ \ \bar{\mathbf{Z}}\bar{\mathbf{y}} = \left(\mathbf{Z}\ \hat{\mathbf{Z}}\right)\begin{pmatrix}\mathbf{y}\\ \hat{\mathbf{y}}\end{pmatrix} = \mathbf{Z}\mathbf{y} + \hat{\mathbf{Z}}\hat{\mathbf{y}}$$

provide an efficient way to update system (3) starting from system (2). Granted the number of new data points *M* is not too high, the new solution $\bar{\mathbf{w}}$ is in general close to the previous one $\mathbf{w}$. Thus, the previous solution can be used as initial iteration of the CG method, which then will find the new solution $\bar{\mathbf{w}}$ within a few iterations (for more details consult Paprotny and Theß 2013). This way, we have constructed an accurate and efficient way for incremental updates of the explicit feature map SVM.

Using these updated SVM models, XELOPRO applies an optimization to define optimal control parameters for a specified situation. First, a subset of the model's attributes is selected as control parameters, i.e., attributes which can be altered or freely selected. For configuring Kanban systems, these are either the number of Kanban containers or the inventory level if the amount of containers is fixed. In addition, an objective function has to be specified, usually either maximizing or minimizing the regression model's output value or combinations of several values. For Kanban systems, the objective function consists of a minimization of inventory levels while at the same time minimizing the number of out-of-inventory events.

Mathematically, we define a subset of *k* attributes $\left(\mathrm{x}_{c,1}, \ldots, \mathrm{x}_{c,k}\right), k \leq d$, as control attributes and in each step where we need to find the optimum values of the control attributes we solve the problem

$$\mathrm{argmax}_{\left(\mathrm{x}_{c,1},\ldots,\mathrm{x}_{c,k}\right)} f(\mathbf{x})$$

where the values of all non-control attributes are the original ones. For the feature-map SVM in our Kanban problem, we arrive at

$$\mathrm{argmin}_{\left(\mathrm{x}_{c,1},\ldots,\mathrm{x}_{c,k}\right)} \mathbf{w}^{\mathrm{T}}\mathbf{z}(\mathbf{x})\ . \tag{4}$$

Unlike the learning and application of the SVM function $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{z}(\mathbf{x})$, the optimization problem (4) is nonlinear since $\mathbf{z}(\mathbf{x})$ is non-linear. In general, it is also not convex. Even more, in practical application problem (4) is solved under constraints applying to the control attributes. Thus, we arrive at a non-linear non-convex constrained optimization problem that requires advanced optimization algorithms.

Luckily, in most applications, the number of control attributes *k* is not high; for the presented Kanban case, it is just one. For many applications, as for the Kanban case, the constraints are simple box constraints. Since in the presented case it holds that $k = 1$, the box constraints are just an interval. XELOPRO uses a BFGS optimization algorithm (Fletcher 1987) to solve (4) under box constraints.

In general, for non-linear problems like (4) we cannot guarantee global convergence. However, we can exploit the smoothness of $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{z}(\mathbf{x})$, which is infinitely differentiable. This allows for applying sparse grids to approximate $f(\mathbf{x})$. The sparse grid method is a numerical discretization technique for multivariate problems (Zenger 1991). Sparse grids can be considered as high-dimensional wavelets over anisotropic grids. Especially, sparse grids allow for approximating smooth functions in high dimensions efficiently. In XELOPRO, sparse grids are used to find the region auf the global minimum of (4) and then the solution is refined by applying locally convergent methods like BFGS. This way, problem (4) can be reliably solved.

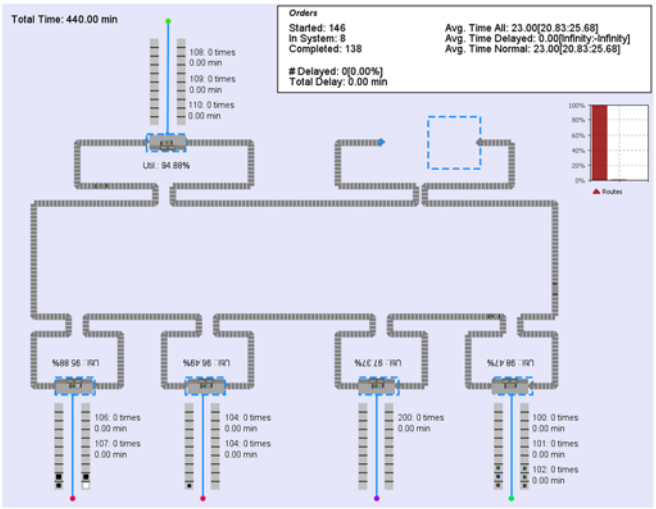
The XELOPRO was originally developed and this far only applied for eCommerce applications, e.g., to estimate sales and customer buying behaviors in online shops. Nevertheless, due to its ability to refine the regression models incrementally at runtime without the need of storing or accessing large amounts of

historical data, it provides a promising tool for a decentralized, continuous reconfiguration of Kanban control systems. Therefore, it is adapted to optimize each products' inventory level based on the current state of the overall production system. The use of regression techniques allows for including additional context information, e.g., times, dates, the actual throughput and demand levels, or even production plans if these are available. As a result, the use of the online learning mechanism bridges the gap between the two classes of dynamic Kanban configuration methods presented earlier. The demand forecasts are replaced by a learnt, adaptive model, whereby the actual optimization monitors the current state of the system and includes additional context information, e.g., about periodic trends within its model.

4 SIMULATION STUDY

To evaluate the online learning approach, a simulation study was conducted. An assembly line for optical sensors was modeled using the AnyLogic process modelling library (Figure 1a). The model comprises the described online learning approach and an implementation for a continuous (time discrete) adjustment of stock levels using a standard Kanban formula. The objective of the simulation study is to benchmark the online learning to the continuous adjustment under different dynamics described later in this section.

a)



b)

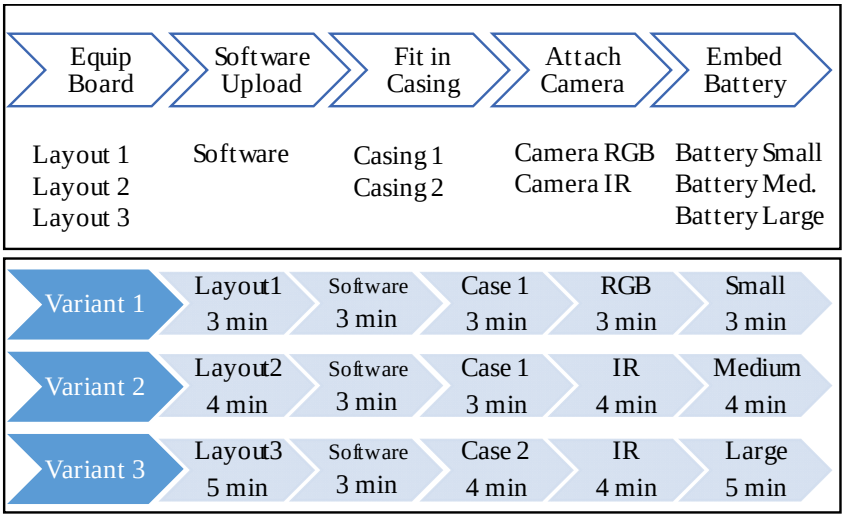


Figure 1: (a) Process chain and product variants (b) Screenshot of the AnyLogic scenario.

4.1 Structure of the Simulation Model

The simulation scenario covers five assembly steps, whereby three different variants of the optical sensor can be manufactured. Each variant follows the same process chain, but consists of different materials. Figure 1b depicts the overall process chain and the available materials in the upper part. The lower part of the figure shows the three different product variants, including the materials and processing times. The model assumes these times as constant. Moreover, each process step refers to an assembly station. In addition to performing the assembly, these stations consist of a local storage for the corresponding materials.

In general, the simulation model uses ConWip (Jaegler et al. 2017; Spearman et al. 1990) to control the order release. Thereby, a predefined amount of orders is allowed within the production system. When an order is finished and leaves the assembly line, a new order is generated and released into the system. Each order consists of a single product referring to a specified product variant. The simulation model determines the variant randomly during order generation, following defined probabilities for the composition of orders, i.e., variants $p(v_1), p(v_2)$ and $p(v_3)$.

In contrast, the local storages use a 2-Card-Kanban system to control their inventory levels. Therefore, each material m is associated to a maximum inventory level inv_m. When half of the inventory is consumed (i.e., one of the two Kanban containers is empty), the assembly station automatically orders a new container by issuing a restock request. Each restock request requires a certain amount of time rt_m until the new container arrives at the assembly station. Depending on the composition of variants, the simulation model records the actual consumption of materials as ac_m and the average throughput time over all orders as th. Overall, the objective is to minimize inventory levels and the number of (material) shortage events $\#OoI_m$ or, respectively, waiting times induced by missing materials OoI_m. Table 1 summarizes the most important model parameters.

Table 1: Parameters of the simulation model.

Description	Parameter
Probabilities for product variants	$p(v_1), p(v_2), p(v_3)$
Throughput time	th
Materials	m
Inventory levels for each material	inv_m
Restock time (time between order and arrival)	rt_m
Average amount of required materials	ac_m
Number of out-of-inventory events	$\#OoI_m$
Waiting time due to out-of-inventory events	OoI_m

4.2 Scenarios and Dynamic Influences

For the evaluation, different simulation scenarios are created, inducing different forms of dynamics to the simulation model. Each scenario is characterized by a constant total production load over a period of three weeks. During this time, the composition of product variants changes depending on the scenario. In addition, the restock time for all materials varies between 39 and 18 minutes per day across all scenarios, whereby restock times are longer in the morning than in the evening (Figure 2). This behavior simulates a characteristic often observed in practical applications, where the central storage usually is resupplied by the beginning of the day or shift and only operates at lower capacities afterwards.

While the changes in restock time are applied to all scenarios, four different scenarios have been defined, applying different trends for the product variant composition as given in Figure 3.

The first scenario does not include any dynamics related to the composition of orders and acts as a basic scenario to compare the efficiency of the Kanban configuration. Therefore, the probabilities that a certain

type of order is generated are set to 50%, 30%, and 20%. Nevertheless, the random generation of orders still introduces a comparably high amount of dynamics as the actual composition over time still varies (Figure 4). The second scenario additionally introduces linear trends, where the probabilities for the variants 1 and 3 decline and the probability for variant 2 increases over time. The third scenario uses a periodic change of the composition, at which the probabilities for variants 1 and 3 vary periodically, whereas the probability for variant 2 fills in the gaps. The fourth scenario describes a stepwise parameter variation. The probability for variant 1 is decreased in steps of 20% whereby the probabilities for the other variants are permutated to fill in the remaining probabilities also in steps of 20%.

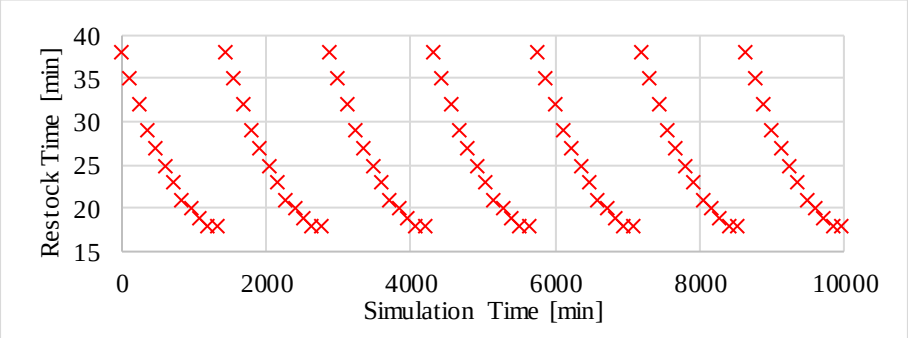


Figure 2: Average restock times for one week, which is repeated for every successive week.

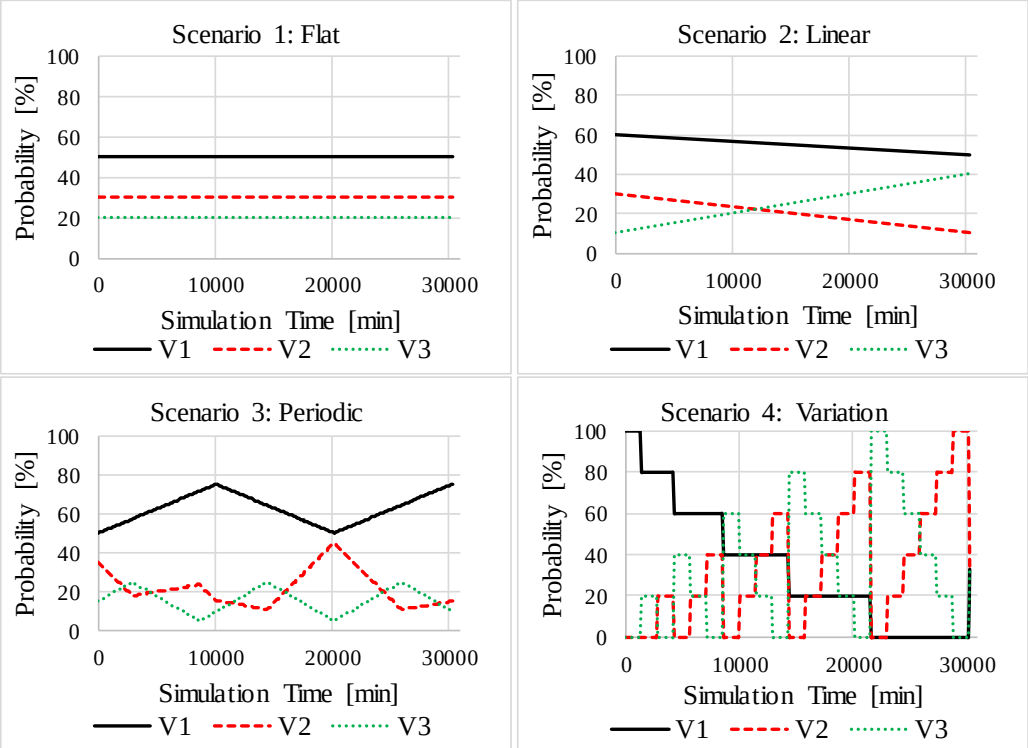


Figure 3: Probabilities for the composition of orders for each of the four scenarios.

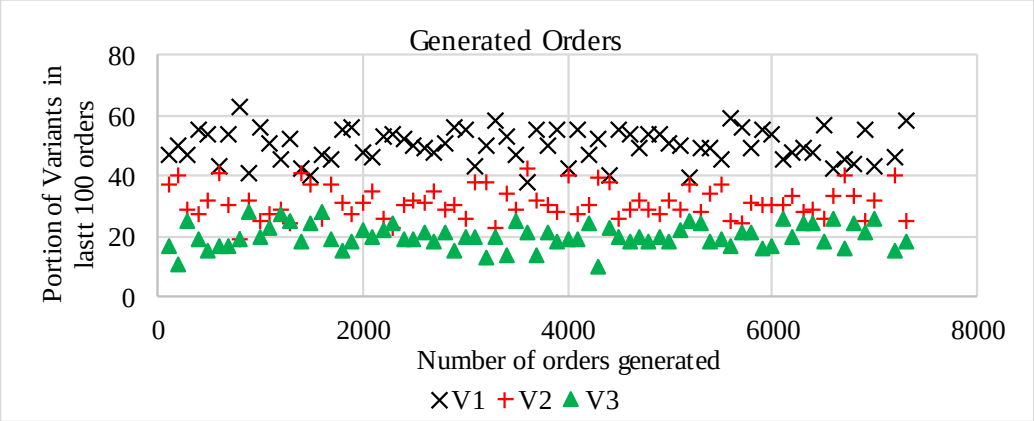


Figure 4: Example for the composition of orders generated in scenario 1.

4.3 Inventory Adaptation

Due to the dynamics described before, a simple *preconfiguration* of the Kanban system in terms of the inventory levels for each material does not provide a sufficient benchmark, but instead poses as worst-case configuration. This results from the fact that such a configuration would have to be able to cope with the maximum amount of consumed materials per period to avoid out-of-inventory events. In cases where this consumption rate fluctuates or changes strongly, the system still retains these high inventory levels. According to (SAP 2018), the simulation model includes a method to update the inventory levels using

$$inv = 2 * \left(\frac{rt * ac}{k - c} * sf\right)$$

by applying a *sliding time window* of 360 minutes of simulation time as a benchmark. Every hour, the inventory level inv is calculated for every material using the last six hours' average consumption rate ac and the current restock time rt. The amount of Kanban containers k is selected as 2; the normalization constant c is selected as 0 due to the small amount of containers. In contrast, the safety factor sf is chosen as 1.5 to compensate for the low consumption rate. The time window of six hours was chosen as a tradeoff between responsiveness and variance. Larger windows reduce the responsiveness to changes in the order composition, while smaller windows are subject to stronger influences due to the random order generation. During the simulation runs, inventory levels are updated in fixed intervals of one hour of simulation time.

The XELOPRO uses a comparable set of information for training. The regression model tries to predict the value of the objective function for a provided material m, at a given time t for the current throughput th, material consumption ac_m, restock time rt_m and inventory level inv_m according to the equation $obj^* = f(t, m, th, ac_m, rt_m, inv_m)$. This input vector consists of all locally available values already used for the classic approach, but additionally includes the throughput time, as it is highly influenced by out-of-stock events and, therefore, prevents the model from selecting too small inventory levels. The current values are again sourced using a six hour time window. The inclusion of production plans or other information available to central planning systems has been omitted to achieve a high degree of decentralization for the Kanban control. During the optimization, XELOPRO tries to select the inventory levels to minimize the regression model. In order to train the model, the objective function has been selected to consist of the materials inventory level inv_m, the throughput th, and the waiting time OoI_m induced by out-of-inventory events as $obj = inv_m + th + OoI_m$.

Thus, the SVM uses six attributes (d = 6) where the inventory level inv_m serves as control attribute. The control attribute values are restricted to the interval [0, 30]. To ensure a high degree of decentral control, each inventory is assigned its own XELOPRO. Thus, each model is trained and evaluated independently from the others. To decrease the ramp-up time, the regression model is pretrained using data from two

simulation runs (six weeks of simulation time). The interval of six weeks was chosen as the system already showed good performance after three weeks, but became more performant after a ramp-up time of six weeks. Further pretraining did not show a significant improvement. During execution, the simulation model provides data for the online training every ten minutes of simulation time, whereby the inventory levels are adjusted every hour using the built-in optimization.

4.4 Results

Table 2 summarizes the results of the simulation runs. For each scenario and each method, the table lists the sum over the average inventory levels ($\sum \emptyset inv_m$), the number of out-of-inventory events ($\#OoI_m$), the average of active waiting times over three weeks of simulation time ($\emptyset\ OoI_m$) and the amount of orders finished in total (Finished) and per hour (Per Hour). As the preconfiguration provides a baseline benchmark for the maximum required inventory levels for each scenario, these levels were selected by incrementally increasing the inventory level until no out-of-inventory events occurred during 10 replications. While the preconfiguration provides the worst-case for inventory levels, it also constitutes a best-case benchmark for the throughput and finished orders values for each scenario. In contrast to the static configuration, the sliding window and online learning approaches tried to optimize the inventory levels during execution. As a result, Table 2 additionally states the number of delayed orders as an absolute value (Delayed) and as a relative value with respect to the number of completed orders (% Delayed), each for a single repetition for each scenario. Nevertheless, multiple repetitions showed similar results due to the high number of orders within the simulated timeframe.

Table 2: Simulation results.

		S1: Flat		S2: Linear		S3: Periodic		S4: Variation	
Preconfigured	$\sum \emptyset inv_m$	178		184		192		218	
	Finished	8138		8121		8424		7630	
	Per Hour	16.15		16.11		16.71		15.14	
Sliding Window	$\sum \emptyset inv_m$	84.47		84.81		88.09		81.07	
	$\#OoI_m$	608		632		552		875	
	$\emptyset\ OoI_m$	188.19		165.61		151.20		313.93	
	Finished	7624		7647		7951		6969	
	Per Hour	15.13		15.17		15.78		13.83	
	Delayed	498		499		444		645	
	% Delayed	6.53		6.53		5.58		9.26	
Online Learning (relative values in comparison to the sliding window approach)	$\sum \emptyset inv_m$	92.26	+9.2%	93.00	+9.7%	92.72	+5.3%	106.50	+31.4%
	$\#OoI_m$	410	–32.6%	427	–32.4%	464	–15.9%	343	–60.8%
	$\emptyset\ OoI_m$	104.89	–44.3%	165.69	0.0%	143.64	–5.0%	138.35	–55.9%
	Finished	7763	+1.8%	7729	+1.1%	8062	+1.4%	7325	+5.1%
	Per Hour	15.40	+1.8%	15.34	+1.1%	16.00	+1.4%	14.53	+5.1%
	Delayed	381	–23.5%	373	–25.3%	418	–5.9%	288	–55.3%
	% Delayed	4.91	–24.9%	4.83	–26.0%	5.18	–7.2%	3.93	–57.5%

Both of the dynamic approaches heavily reduce the overall inventory level compared to the preconfiguration. In general, the results show a small increase in the average inventory levels for the online learning approach in comparison to the sliding window approach. For the simple scenarios 1–3, these range from ~5% to ~10%. For the fourth scenario, the online learning approach selects the inventory levels

approximately 31% higher than the sliding window. As a result, the amount of out-of-inventory events decreases strongly, between 16% and 32% for the simple scenarios, and by about 60% for scenario four. The overall amount of delayed orders reduces proportionally. As a result, the amount of orders finished during the three-week simulation time increases for the online learning approach by 1% to 2% for scenarios one to three and approximately 5% for scenario four. The low decrease in finished orders can be explained by the high base utilization. The ConWip is configured to achieve approximately 95%–99% utilization on the first assembly station, which also constitutes the bottleneck machine, if no delays occur. Thereby, the utilization is also subject to the composition of variants, as e.g., variant 3 takes longer to manufacture. In general, the online learning approach reduces the occurrence of out-of-inventory events strongly, by increasing the average inventory levels slightly compared to the sliding window approach.

Figure 5 depicts the inventory levels for the material “Layout 1” which is only used as part of the first product variant both for the online learning approach and for the sliding window. In addition, the graphs show the probability for the first product variant for comparison. Both approaches show variations in the proposed inventory levels, which can be accounted to the dynamics introduced by the random generation of orders and the length of the time windows used to record the current material requirements. In general, the online learning approach achieves a slightly smoother shape and maintains a slightly higher inventory level, resulting in fewer out-of-inventory events. The same characteristics can be observed for the inventory levels of other materials across all simulation scenarios.

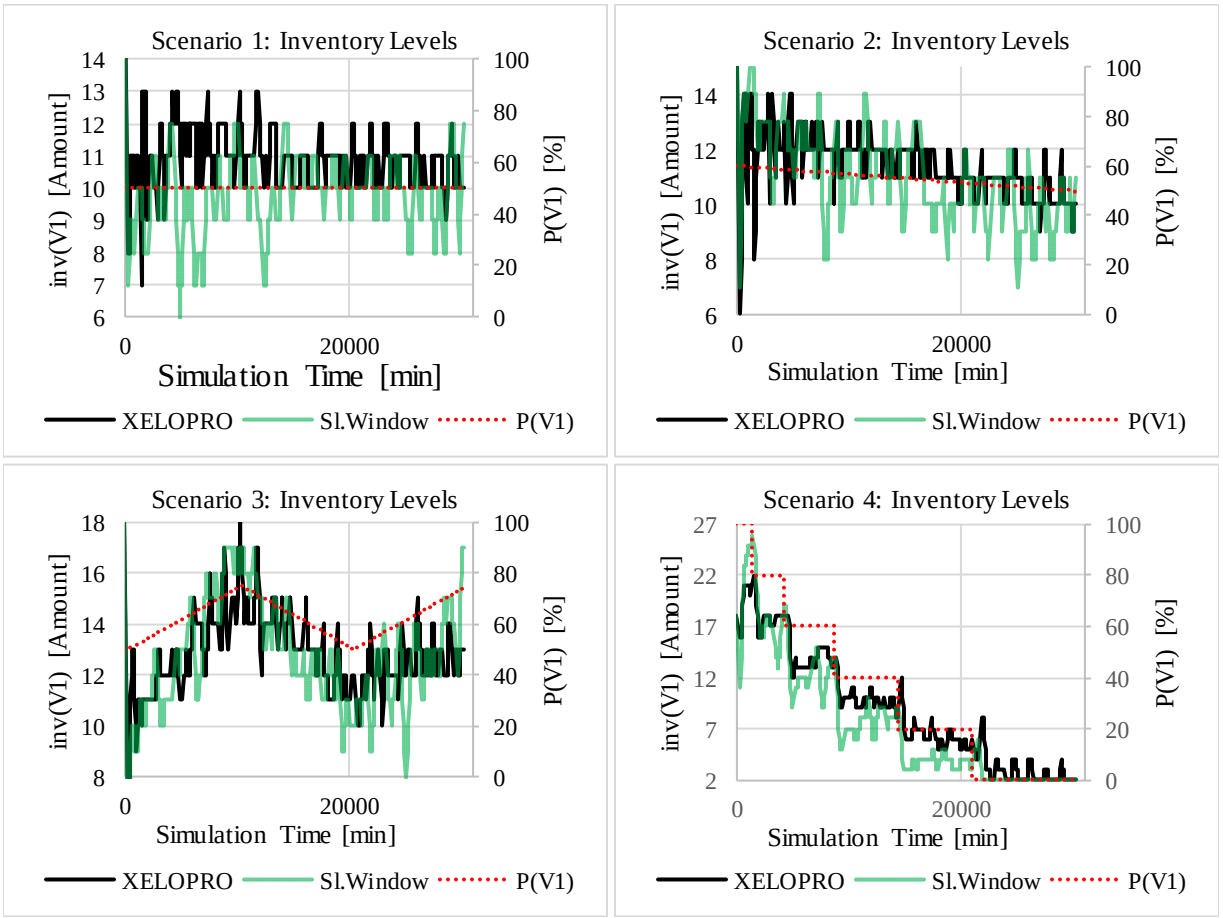


Figure 5: Inventory levels for the material "Layout 1" for the first product variant in all scenarios.

5 CONCLUSION AND FUTURE WORK

This paper aims at investigating the application of online learning as an alternative method for the decentralized configuration of Kanban pull controls for inventory management in variant-rich assembly scenarios. Therefore, the XELOPRO online learning platform is used to continuously train a specialized support vector machine, which is used to optimize the inventory levels. Results of the conducted simulation study show a high potential for the use of online learning to adapt the inventory levels of local storages dynamically at runtime. Thereby, the online learning approach does not rely on centralized information, so that each inventory can be optimized independently from other inventories or a central planning tool to achieve a fully decentralized control. Compared to dynamically applying standard Kanban equations, online learning reduces the amount of delayed orders by an average of about 25% across all tested scenarios. In comparison to a static configuration of the respective Kanban systems, the approach achieves a drastic reduction of the average inventory levels by approximately 50%.

The highly decentralized nature of the proposed approach enables a high scalability to arbitrarily complex production systems. As each workstation uses its own local model, interdependencies between different stations do not have to be included explicitly but are implicitly covered by the recorded material consumption. Moreover, the proposed approach can easily be applied to systems, in which the production itself uses a Kanban control instead of the described ConWip push approach. Basically, products coming from an earlier production stage can be considered materials as well. In such cases, the restock times for that particular material would be longer as it has to be produced and would probably exhibit stronger dynamics than the restock times assumed in this paper (compare Figure 2). Nevertheless, it can be assumed that the respective SVM model would adopt comparably quickly to these dynamics.

Future work will focus on refining the used pool of data. For this paper, a simple objective function including the current material consumption, throughput, and the number of out-of-inventory events has been used for the optimization of inventory levels. A more sophisticated objective function will probably lead to an even better performance. Nevertheless, this objective function already covers the most important aspects, i.e., minimizing the inventory and out-of-stock events. As a result, it can be expected that the throughput, which is strongly dependent upon the number of out-of-stock events, can be omitted by further weighting these events against the stock levels. In addition, the online learning only relies on the current state of the system. Enforcing a prediction of future demands based on the learnt models can further increase the performance, especially in scenarios with periodic or recurring patterns in customer behavior. At this stage, the approach has been designed to omit the use of centralized information, e.g., of production plans. Future research will additionally investigate the effects and restrictions of using such centralized information in model building.

6 REFERENCES

Belisário, L. S., and H. Pierreval. 2015. "Using Genetic Programming and Simulation to Learn How to Dynamically Adapt the Number of Cards in Reactive Pull Systems“. *Expert Systems with Applications* 42:3129–3141.

Fletcher, R. 1987, *Practical Methods of Optimization*. 2nd ed. Chichester: John Wiley & Sons.

Framinan, J. M., P. L. González, and R. Ruiz-Usano. 2006. "Dynamic Card Controlling in a Conwip System". *International Journal of Production Economics* 99:102–116.

Gupta, S. M., and Y. A. Y. Al-Turki. 2010. "An Algorithm to Dynamically Adjust the Number of Kanbans in Stochastic Processing Times and Variable Demand Environment". *Production Planning & Control* 8:133–141.

Jaegler, Y., A. Jaegler, P. Burlat, S. Lamouri, and D. Trentesaux. 2017. "The ConWip Production Control System: A Systematic Review and Classification“. *International Journal of Production Research* 33:1–22.

Kimura, O., and H. Terada. 2007. "Design and Analysis of Pull System, a Method of Multi-stage Production Control“. *International Journal of Production Research* 19:241–253.

Korugan, A., and S. M. Gupta. 2014. "An Adaptive CONWIP Mechanism for Hybrid Production Systems". *The International Journal of Advanced Manufacturing Technology* 74:715–727.

Majava, J., and T. Ojanperä. 2017. "Lean Production Development in SMEs: A Case Study". *Management and Production Engineering Review* 8:481.

Ohno, T. 2013, *Das Toyota-Produktionssystem*. 3rd ed. Frankfurt, New York: Campus.

Paprotny, A., and M. Theß. 2013, *Realtime Data Mining. Self-learning Techniques for Recommendation Engines,* 2nd ed., Basel: Birkhäuser.

Parente, M., U. B. Schimpel, and S. Vajjala. 2012. "Patent Application US20120136758A1: Production Management with Multi-type Kanban Modeling", accessed April 4th, 2018. https://patents.google.com/patent/US20120136758.

Rahimi, A., and B. Recht. 2008. "Random Features for Large-scale Kernel Machines". *Advances in Neural Information Processing Systems 20*:1177–1184.

Rees, L. P., P. R. Philipoom, B. W. Taylor, and P. Y. Huang. 1987. "Dynamically Adjusting the Number of Kanbans in a Just-in-Time Production System Using Estimated Values of Leadtime". *IIE Transactions* 19:199–207.

SAP. 2018. *Ermitteln Anzahl Kanbans/Kanbanmenge*. https://help.sap.com/saphelp_di46c2/helpdata/DE/cb/7f8c3943b711d189410000e829fbbd/frameset.htm, accessed April 4th, 2018.

Shahabudeen, P., K. Krishnaiah, and M. Thulasi Narayanan. 2003. "Design of a Two-Card Dynamic Kanban System Using a Simulated Annealing Algorithm". *The International Journal of Advanced Manufacturing Technology* 21:754–759.

Signal Cruncher GmbH. 2018. *XELOPRO*. https://signal-cruncher.com/, accessed April 4th, 2018.

Sivakumar, G. D., and P. Shahabudeen. 2008. "Design of Multi-stage Adaptive Kanban System". *The International Journal of Advanced Manufacturing Technology* 38:321–336.

Spearman, M., D. Woodruff, and W. Hopp. 1990. "CONWIP: A Pull Alternative to Kanban". *International Journal of Production Research* 28:879–894.

Takahashi, K., K. Morikawa, D. Hirotani, and T. Yoshikawa. 2010. "Adaptive Kanban Control Systems for Two-stage Production Lines". *International Journal of Manufacturing Technology and Management* 20:75–93.

Talibi, Z., H. Bril El Haouzi, and A. Thomas. 2013. "The Relevance Study of Adaptive Kanban in a Multicriteria Constraints Context Using Data-driven Simulation Method". In *Proceedings of 2013 International Conference on Industrial Engineering and Systems Management (IESM)*, edited by D. Aboutajdine et al., 169–175. New-York: IEEE.

Tardif, V., and L. Maaseidvaag. 2001. "An Adaptive Approach to Controlling Kanban Systems". *European Journal of Operational Research* 132:411–424.

Zenger, C. 1991. "Sparse Grids". In *Parallel Algorithms for Partial Differential Equations,* edited by H. Hackbusch, 241–251. Braunschweig: Vieweg.

Zhang, L., Z. Wang, B. Wu, and J. Li. 2015. "The Study and Development of E-kanban Management in MES for Hardware Plastics Production Workshop". In *Proceedings of the 2015 International Conference on Intelligent Systems Research and Mechatronics Engineering (ISRME 2015)*, edited by J. Liu et al., 322–329. Amsterdam: Atlantis.

AUTHOR BIOGRAPHIES

DANIEL RIPPEL is research scientist at BIBA – Bremer Institut für Produktion und Logistik GmbH at the University of Bremen. He holds a Diploma degree in Computer Sciences from University of Bremen. His research interests include modelling and simulation of logistic systems, the development of domain-specific modelling methods, as well as the application of prediction techniques from the areas of statistics and machine learning. His e-mail address is rip@biba.uni-bremen.de.

MICHAEL LÜTJEN heads the department Data Analytics and Process Optimization at BIBA – Bremer Institut für Produktion und Logistik GmbH at the University of Bremen. He holds a Master's degree in Industrial Engineering from the University of Applied Sciences in Wilhelmshaven with focus on 'Simulation and optimization in production'. His research interests include modeling and simulation of complex production scenarios. His e-mail address is ltj@biba.uni-bremen.de.

MICHAEL THESS is managing director of Signal Cruncher GmbH, a spin-off of the real-time analytics vendor prudsys AG. He studied mathematics in Chemnitz und St. Petersburg. As one of the founders of prudsys he was responsible for research and development. He is specialized in numerical analysis and multilevel approximation theory, especially in the context of reinforcement learning. His e-mail address is thess@signal-cruncher.com.

MICHAEL FREITAG is Full Professor at the University of Bremen and Director of BIBA – Bremer Institut für Produktion und Logistik GmbH at the University of Bremen. He holds a Diploma degree in Electrical Engineering and a Doctoral degree in Production Engineering. His research interests include modelling, simulation, and optimization of complex production and logistics systems, the development of planning and control methods for logistic processes, and the automation of physical material flows through robots and flexible transport systems. His e-mail address is fre@biba.uni-bremen.de.