Optimizing real-world factory flows using aggregated discrete event simulation modelling

Creating decision-support through simulation-based optimization and knowledge-extraction

Simon Lidberg¹ · Tehseen Aslam¹ · Leif Pehrsson¹ · Amos H. C. Ng¹

Published online: 1 July 2019
© The Author(s) 2019

Abstract

Reacting quickly to changing market demands and new variants by improving and adapting industrial systems is an important business advantage. Changes to systems are costly; especially when those systems are already in place. Resources invested should be targeted so that the results of the improvements are maximized. One method allowing this is the combination of discrete event simulation, aggregated models, multi-objective optimization, and data-mining shown in this article. A real-world optimization case study of an industrial problem is conducted resulting in lowering the storage levels, reducing lead time, and lowering batch sizes, showing the potential of optimizing on the factory level. Furthermore, a base for decision-support is presented, generating clusters from the optimization results. These clusters are then used as targets for a decision tree algorithm, creating rules for reaching different solutions for a decision-maker to choose from. Thereby allowing decisions to be driven by data, and not by intuition.

Keywords Aggregation · Discrete event simulation · Multi-objective optimization · Industrial case study · Data mining · Decision support

✉ Simon Lidberg
simon.lidberg@his.se

Tehseen Aslam
tehseen.aslam@his.se

Leif Pehrsson
leif.pehrsson@his.se

Amos H. C. Ng
amos.ng@his.se

¹ School of Engineering Science, University of Skövde, Högsålevägen, Box 408, Skövde, Sweden

1 Introduction

The manufacturing industry in general, and the automotive industry in particular, has entered a stage of major change and transformation. Continuously shorter product life-cycles with reduced implementation times, an increasing number of variants, and new technologies e.g.; autonomous cars, electrification, and ride-sharing, are positioned to shape the industry. Current and future production systems will need to be more flexible and reconfigurable to meet these challenges. At the same time the lead time will need to be reduced while keeping internal and external customers sated (Koren 2010). Higher quality decision-support, delivered faster than before, is one way of supporting companies to overcome these challenges. For production systems, with a focus on production lines, the established way of making predictions about future performance is using discrete-event simulation (DES) models (Negahban and Smith 2014).

During the manufacturing system lifecycle—beginning in the concept phase and ending in the decommissioning of the system—a valid DES model can provide decision-support in all phases. If the same model is used, and expanded, to represent the different stages of the system, the complexity of the model is determined by the most detailed question put to the model (Chwif et al. 2000, 2006). The analysis of manufacturing systems and factories can be performed from different perspectives and level of detail. Jain et al. (2015) argues for a multi-resolution modelling approach to a complete virtual factory, incorporating different types of modelling for each different level, spanning through device, station, cell, and department culminating in the factory level while describing their view of the virtual factory.

Different simulation aggregation levels can be referred to as microscopic, mesoscopic and macroscopic (Toledo et al. 2010; Reggelin and Tolujew 2008; Hennig et al. 2014). Macroscopic modelling involves a high degree of aggregation of data, processes and functionality, analyzing the general behavior of the system. The individual entities are not important at the macroscopic level, instead, they are aggregated as flows constituting the smallest entity of interest. The advantage of modelling at the macroscopic level is the low development time and the low computational cost for running the model (Burghout 2004). On the other end lies the microscopic level, where every individual entity is modeled with an own set of governing rules and behaviors. The microscopic level offer high detail and good validity, but are time consuming to build and run. Between these levels lies the mesoscopic level where processes and sub-systems deemed crucial are modelled in detail, with the intention of aggregating as much as possible of the other processes, resulting in a trade-off between computational time and construction time. In mesoscopic simulation of logistic systems, the advantages are also a high degree of flexibility in preparing input data and a simpler structure of the model data (Reggelin and Tolujew 2008).

Frantz (2003) has developed a taxonomy for describing different abstraction techniques in DES models, each reducing complexity by modifying the model boundary, model behavior, or model form. Another classification from Chwif et al. (2006) divides the simplification techniques in three categories: omission, aggregation, and

substitution. Reducing model complexity by aggregation has been studied in previous work by Pehrsson et al. (2015), and the same mesoscopic technique has been applied to manually experiment with safety levels to reduce storage levels (Lidberg et al. 2018a). Reducing both lead time and storage sizes while maintaining a high delivery precision reduces the cost of inventory and time to market and is of great importance for many companies (Hopp and Spearman 2011). Utilizing multi-objective optimization has the potential to move key parameters closer to their ideal objective values, but also enables new insights to be gained about the behavior of the industrial system. The process of improving simulation models, and in extension real physical systems, combining DES and multi-objective optimization is commonly referred to as simulation-based multi-objective optimization (SMO).

This article, an extended version of Lidberg et al. (2018b), begins with a background section exploring discrete event simulation, aggregation techniques, multi-objective optimization and data-mining. Using these concepts, an aggregated model will be constructed for an industrial system to be used in an optimization setup. By combining the aggregation techniques and multi-objective optimization in a real world industrial case study enables lowering of the storage levels by 31%, reducing lead time by 67%, and lowering batch sizes by more than 50%, showing the potential of optimizing on the factory level. Further analysis of the results can be made using data-mining and k-means clustering where insights from the optimization could be used to improve the industrial system. Finally, the article ends with a discussion of the approach and results, conclusions, and suggestions for future work.

2 Background

Aggregated modelling can enable faster optimization results and therefore faster decision-support. The first part of this section deals with simulation, modelling, and the possibility of reducing the computational time, data collection time and time required for modelling by aggregating data and logic. Secondly, the chosen aggregation technique is described in more detail, with examples and validation for the usage of the technique. Finally; the third part summarizes how optimization using evolutionary genetic algorithms can be applied to find solutions to multi-objective problems.

2.1 DES and aggregated modelling

Building models to represent an already existing physical system or planned system is now common practice in many industries (Negahban and Smith 2014; Banks et al. 2009). The models are used to improve or verify system solutions for either future or current production lines. In early project phases before a physical system exists, constructing models is the best approach to predict future performance. Testing proposed changes in models before implementing them physically increases the confidence and robustness of the proposed solution and reduces the likelihood of implementing costly mistakes.

Models should be built with a purpose, usually to answer one or several specific questions. Each different question requires different levels of model detail and have different requirements for input data. An in-depth or difficult question could trigger the development of several sub-systems with each requiring additional input data. Adding detail often means that the computational time will increase when running the model, but also that the simulation project duration will be extended for gathering data and verifying model correctness, a cost incurred during every model update (Chwif et al. 2000; Fowler and Rose 2004).

Reducing the delivery time of decision support to the decision maker one approach is reducing the level of details in the models and thereby reducing the computational complexity. This decreases time needed to complete an optimization study and therefore delivers decision support faster to a decision maker. Both analytic and hybrid modelling solutions have been developed to aggregate model detail (Zulkepli et al. 2012; Lefebvre and Armbruster 2011; Asmundsson et al. 2006). Other approaches to simulation modelling aggregation, such as aggregating by replacing the complex model with a System Dynamics model or a neural network, are developed and applied to manufacturing and logistic problems by Oyarbide et al. (2003) and Kuo et al. (2007).

With aggregation through DES modelling, reducing the requirements of input data, and while maintaining the output characteristics of a detailed model, one approach has been developed by Pehrsson et al. (2014) and Pehrsson et al. (2015). One advantage of this method is the possible combination of existing models and aggregated models, because of the basic component being a discrete entity. The entities can transfer from an aggregated predecessor, to a detailed DES model, and finally to an aggregated successor in the same study if certain behaviours require the detailed model to remain in the detailed state.

Using this aggregation technique, the input parameters required for the aggregation of production lines are processing time, availability, mean down time, average work in process, *AvgWIP*, maximum work in process, *MaxWIP*, and minimum lead time. The processing time, availability and mean down time are only required to be gathered for the first and last station in the system. This enables faster model development while retaining the possibility of connection and extension with other DES objects. In the next section, an implementation of this technique in a discrete event simulation software will be discussed.

2.1.1 Implementation of the aggregation technique

Five objects are needed to implement the aggregation technique, shown in Fig. 1. The objects *LineInput* and *LineOutput* governs most of the variability and stochasticity in the aggregation technique. These objects are joined by a buffer object called *LineWIP*, where the lead time is represented in the model. Controlling the WIP is handled by a pallet system where the total number of pallets is set by the *MaxWIP* parameter. At *LineInput* the WIP pallets are assembled with incoming products which are transported through the system. When exiting *LineOutput*, the pallets are passed to *WIPControl* and the products are sent to the next step in the process. The

Fig. 1 Schematic description of the aggregation technique

pallet system is similar in principle to a CONWIP system by Spearman et al. (1990) with a delay on the return of the pallets to *LineInput*.

The *WIPControl* object returns pallets to *LineInput* according to an exponential distribution based on the parameters of the model. The *WIPControl* is configured as a parallel station, with as many positions as the parameter *MaxWIP* states, meaning that for each pallet return, from *LineOutput* to *LineInput*, the expected mean time in seconds is $E[X] = \beta$ as defined in Eq. 1. The delay function is calculated based on Little's law (Little 1961). The objective of the *WIPControl* is to balance, or control, the *WIP* value in *LineWIP* towards an determined expected average *AvgWIP*.

$$f(x;\beta) = \begin{cases} \frac{1}{\beta} e^{-\frac{x}{\beta}} & x \geq 0, \\ 0 & x < 0 \end{cases} \quad \text{where} \quad (1)$$

$$\beta = \frac{(\text{MaxWIP} - \text{AvgWIP}) \times \text{ProcessingTime}}{\text{Availability}}$$

2.1.2 Aggregation technique with variant based setup times

Setup functionality has been added to the *LineInput* and *LineOutput* objects in the aggregation technique. This is an addition to the technique originally described by Pehrsson et al. (2015). Changeover times in the objects enable dynamic variability based on the production planning of the orders entering the system. Compared with analyzing and adapting failures to fit the changeovers, using setup functionality increases the generalizability of the technique. Setup between product variants is an important functionality to model particular lines, one example of such a line is included in the case study for this article. The changeover times between variants are defined in a matrix which allows for differentiated times; e.g. time taken for changing Var_A to Var_B could differ from changing Var_B to Var_A .

To validate the addition of setup times, three models are built; (1) *Aggregated*, the original aggregation technique, (2) *Aggregated Setup*, the aggregation technique with the additional setup functionality, and (3) *Detailed*, the detailed model which the aggregated models are based upon. The aggregated models share the same input parameters, with the addition of setup time to the *Aggregated Setup* model, shown in Table 1. In the *Detailed* model, setup times are individual for the different machines, but changing from one variant to the next, the impact on

Table 1 Setup time, in minutes, between product variants for the *Aggregated Setup* model

	Var_A	Var_B	Var_C
Var_A	—	45	45
Var_B	45	—	45
Var_C	45	45	—

the line results in the times shown in the changeover matrix. The *Detailed* model has approximately 40 machines, grouped in 29 operations, meaning that some are configured as parallel machines while the rest are sequential.

The variants enter the model in differently sized batches in a cyclical sequence; Var_A 2348 products, Var_B 1210 products, and Var_C 864 products. Although the setup times are identical between the variants, the batch size difference will affect the disturbance to the line caused by the changeover. In Fig. 2, output from the three different models are shown and evaluated by three key performance indicators (KPI). The three *KPI*'s used to evaluate the performance are jobs per hour (JPH), leadtime (LT), and work in process (WIP); these are summarized by mean value, median value, minimum value, and maximum value in Table 2.

Analyzing the trends in Fig. 2, *Aggregated Setup* more closely resemble *Detailed* compared to the original technique. Approximation of behaviour is the primary concern with this type of modelling, meaning that some lateral shift of the outputs are acceptable. The setup will not initiate during the same time period

Fig. 2 Comparison of model performance with Jobs Per Hour (JPH), LeadTime (LT), and Work in Process (WIP). Shows only a single replication of the models

Table 2 Summarized output from Fig. 2 with LeadTime (LT) measured in hours, Jobs Per Hour (JPH) and Work In Process (WIP). Mean, median, minimum and maximum values are reported

Model	Measurement	Mean	Median	Min.	Max.
Detailed	JPH	73.11	80.00	3.00	114.00
Aggregated	JPH	75.01	80.00	25.00	86.00
Aggregated Setup	JPH	71.48	78.00	1.00	86.00
Detailed	LT	2.97	2.85	1.14	6.77
Aggregated	LT	2.71	2.67	1.50	4.61
Aggregated Setup	LT	3.00	2.95	1.72	5.67
Detailed	WIP	214.94	207	24	443
Aggregated	WIP	201.28	199	123	302
Aggregated Setup	WIP	213.26	216	62	330

in all models due to stochastic differences inherent in *DES* models. The correlation between *Detailed* and *Aggregated Setup* is most visible when comparing by *LT* and *JPH*, showing an increase in *LT* due to the accumulation of parts during setup which matches the detailed model. The setup also affects the *JPH*, lowering the throughput during the changeover until the new variant arrives at the output. Some discrepancies are visible, primarily for *JPH* where the detailed model shows values above 90 *JPH*. This difference can be explained by two causes: variant specific processing times and layout differences.

One variant—out of three in the detailed model—has a lower processing time in relation to the other variants resulting in a higher *JPH* value. The aggregated models are limited to a single processing time due to the incorporation of the processing time in the *WIPControl* delay distribution. This could inhibit the application of the technique to lines where the processing time difference between variants is large. For smaller differences between processing times the aggregation technique matches the output irregardless. The second cause is differences in layout between the detailed model and the aggregated models. The last section of the detailed model operates at a lower processing time for all variants. If a build-up of parts occur in the last section of the line, the throughput rate of the model increases for a short time. This behaviour is also not possible to model using the current aggregation technique.

The experiment results show improved validity through the addition of setup times compared to a detailed model. More effort could be expended to improve the effects of the addition of setup time functionality. Also, the computational time needed for the aggregated model is near constant, while the detailed model require minutes for every evaluation increasing by the complexity of the model. Evaluating the accuracy of the technique in more detail and comparing to a range of different manufacturing systems will be left to further research.

2.2 Simulation-based optimization for many objectives using genetic algorithms

Manual experimentation or design of experiment studies have several limitations. Preparing experiments is time-consuming and an exhaustive iteration through the design parameters will lead to increasingly large experiment setups. This also leads

to larger results, which are difficult to interpret. In the effort to contain the size of the experiment sets, limits are imposed to the experiment inputs, which also impose limits on the insights gained from the results.

Meta-heuristics, such as genetic algorithms, can be utilized to optimize these settings without limiting the inputs (Floreano and Mattiussi 2008). Letting the algorithm, preferably without external bias, explore the search space will lead to better results which are also non-intuitive (Deb 2001). Meaning that these algorithms can develop solutions which are not trivial, or even impossible, to deduce manually.

Genetic algorithms are a sub-class of evolutionary algorithms based on the theory of natural evolution, selecting individual solutions best fitted to the environment, i.e. fulfilling the objectives stated, for further evolution until reaching a termination criteria (Holland 1992). The best solutions, or parents, from each generation are selected and combined, using crossover operators, to create offspring solutions with better chances of attaining higher fitness values. Applying a genetic algorithms to a problem involves six phases; initial population, fitness function, selection, crossover, mutation, and termination (Goldberg 1989).

Determining the size of the population will be the first choice in the setup of the genetic algorithm. Diversity of the initial solutions is very important as to not be trapped in sub-optimal solutions. The size will depend mainly on the properties of the search space and the computational cost of evaluating all the individuals using the fitness function (Floreano and Mattiussi 2008). The fitness function in a SMO involves evaluating a simulation model repeatedly to obtain statistically reliable mean results as outputs. The outputs are then directly mapped as objectives or are combined with other outputs to form objectives to be minimized or maximized. Inputs to the model are encoded into the genome of the individual solutions, where each parameter can be thought of as a gene. When the initial population has been evaluated using the fitness function each individual has been attributed with a fitness score and the selection process can begin.

The selection mechanism determines which solutions to use as the base for the creation of offspring solutions. Genetic material is recombined by the crossover operator with a probability p_c . Crossover operators are designed for different genetic representations, some are only usable when the representation is real-valued, and others recombine sequences. By recombining two good solutions, the idea is that the offspring solutions have a larger chance of being even better. To keep the diversity in the population and avoid stagnating in a local minima of the optimization, each gene of the individual also has a probability, p_m , to be slightly modified by mutation (Floreano and Mattiussi 2008). After creation of the offspring, the new solutions are evaluated using the fitness function to determine their score. Depending on the implementation of the specific algorithm, the population is culled, removing the least effective solutions until the original population size is reached. This process of evaluation, selection, crossover, and mutation ends by fulfilling the termination criteria, i.e. the population improvement rate has either stagnated or ended, or a set number of generations has been reached.

NSGA-II is one example of a multi-objective genetic optimization algorithm which has frequently been applied in SMO due to its applicability, accuracy, convergence and diversification of solutions on the Pareto front (Deb et al. 2002; Ng et al.

2011). Recent research has resulted in the new algorithm NSGA-III which exhibits improved performance for more than three objectives compared to other algorithms (Deb and Jain 2014; Jain and Deb 2014; Li et al. 2015). The NSGA-III algorithm depends on user-supplied or calculated reference points in the objective space to further diversify the solutions. These points are placed on a normalized hyper-plane constructed for all objectives, and for each solution the distance to these points are minimized instead of selection by maximizing the crowding distance to neighbors as in NSGA-II. Selection by proximity to the reference points will increase the diversification of the solutions and thereby increase the breadth of the solutions (Chand and Wagner 2015).

2.3 Knowledge discovery using data-mining

Results from a multi-objective genetic algorithm used in SMO are usually in the tens of thousands where each result have multiple inputs, outputs, and objectives. Analyzing this amount of data is not a trivial task and require additional techniques for extracting knowledge about why certain settings are better than others for some objectives. For multi-objective optimization, visualizing results for more than three objectives is complex. Data-mining techniques are suitable for this purpose, analyzing a large amount of data to find connections and extract rules to further gain knowledge about the system studied (Frawley et al. 1992; Fayyad et al. 1996; Witten et al. 2017).

In Ng et al. (2009) and Dudas et al. (2011), the concept of simulation-based innovization (SBI) is presented. SBI is defined as the application of data-mining techniques, where the knowledge, in the form of decision rules, is captured from the data generated by a SMO process. Four main steps are used (1) *SMO*, (2) *data preprocessing*, (3) *pattern detection* and (4) *interpretation and evaluation*. These are similar to the steps presented in Frawley et al. (1992), but here the data is generated by an optimization process on a DES model and not historical data. SBI utilizes all the evaluations in the results, both Pareto-optimal solutions and non-Pareto-optimal solutions. This choice is made to be able to obtain knowledge about which attributes distinguish Pareto-optimal solutions from non-Pareto-optimal solutions.

To enable the effective use of decision-support to a decision maker, grouping the solutions into clusters, using k-means clustering, can prove valuable. New mean points, or centroids, in the objective space are randomized, which form the base for each cluster, and the solutions are assigned to the cluster with the least squared Euclidean distance in the original form of the algorithm. When all of the solutions are assigned to a cluster, the means are repositioned to the centroid of each respective cluster and the algorithm starts again until there has been no change in the assignment of clusters (Lloyd 1982; Forgy 1965). After obtaining the clusters, or categories, of solutions in the post-optimality phase, a decision tree algorithm can analyze which settings are needed to reach the different regions and rules can be extracted (Dudas et al. 2014).

3 Industrial case study

The following section starts with describing the high level view of the industrial system and how the model—representing the industrial system—is built. A major difference in the functionality between reality and the model is in the scheduling and ordering system implemented, which will therefore be explained in detail. Then, the setup of the optimization will be explained, the different inputs to change, the outputs to measure, and goals to improve. Concluding the section is the results and analysis of the optimization data, both with graphs and data-mining tools.

3.1 Industrial system and model setup

This article uses the same type of industrial setup, an engine manufacturing plant, presented in previous work (Lidberg et al. 2018b). In this section, additional details about the challenges and limitations of the system and model will be presented.

The operational management of the plant has historically been divided into functional groups taking complete care of a specific component being manufactured in the plant. These groups have had the prerogative to design their lines for the introduction of new variants of their components, and also had a large amount of freedom setting the capacities of their production lines with the only requirement reaching a yearly target volume. This has happened without an overarching hand-shake process between the different groups resulting in lines with deviating capacities for the different components.

If the capacity of the production line is insufficient to supply the required demand of the customer, a few options are available; adding and/or improving equipment, or utilizing more production time, i.e. adding additional shifts. If an upstream production line is producing during a shift, but its intended customer is not, the products will have to be stored in the interim. This leads to an increase in the number of products stored and the lead time for the products.

Additional complications arise for certain production lines where a significant time is needed to change tools in the machines when switching to a new variant, known as setup time or changeover time. Mitigating the losses of setup, if the expense of increasing the flexibility of the machines is too large, results in starting a larger amount of a variant—a larger batch—before switching to a new variant. If the customer is not following the same production planning schedule this leads to an increase of stored products until the customer demands the products.

Following a reorganization to a more holistic approach to the plant the operational management is tasked with significantly reducing lead time, the total cost of stored material, and total manufacturing cost. The operational management requested assistance with analyzing the bottlenecks of the system, and reducing storage levels—and thereby lead time—while maintaining delivery precision to the customer. The proposed artefact will need to be able to deliver these outputs while also being re-configurable for repeating the study for different time periods.

Four different component types—each comprised of several variants—are produced by a number of machining lines (ML). Semi-automated assemblies (AA)

Fig. 3 High-level representation of the industrial system and setup of the model

assembles these components into intermediate products which are transported to one of four manual final assemblies (FA), before dispatch to customers. Final good inventories (FGI)—of varying capacities—are separating each of these steps. The inventories store the finished parts from each manufacturing step and re-sorts them into a sequence of products for the downstream process, and ultimately, the customers.

Modelling these interconnected systems in detail would be very computationally expensive to run, and would not allow results to be used for timely decision making. The complete plant is therefore modelled with each production line represented by an aggregated model, using the technique referenced in Sect. 2.1. The largest portion of simulation time is currently the scheduling and ordering logic that is implemented in the model, which is not connected to the aggregation technique.

The setup of the industrial system, but also how the model is constructed, is shown in Fig. 3. The naming scheme for the machining lines consists of ML, with the addition of the component type $\{A, \dots, D\}$, and $\{1, \dots, N\}$ to denote the number of lines for that component. AA and FA are not separated by product type, and only named iteratively, with an approach identical to ML. AA01 and AA02 are conditionally bound to their customer, where AA01 delivers to FA01-02, and AA02 delivers to FA03-04. FGI are named according to the product they store, AFGI stores component A while FAFGI stores complete products. Company participation has only been possible on the condition of obfuscating input and output data from the model to protect company assets. Data are in some cases generalized due to the need for brevity; however, all data relations have been preserved.

One large contributor to the long lead times through the plant is the large batches needed by MLC lines to counteract the setup times required between the different

variants. As measured from the output station, the gap from the last of the leading batch to the first of the trailing batch, is estimated to 45 min of production loss. This estimate is entered in the addition of setup times to the aggregation technique detailed and validated in Sect. 2.1.2.

Ordering and scheduling in the model is facilitated through a safety stock and order point system from the machining final good inventory to AA, detailed in Eq. 2. This calculates the order and safety level for each variant i based on monthly forecasts in the model. In the equation, α represents the safety level in hours and ϕ represents the hourly customer demand. *Leadtime* is the completion time, in hours, for a product from the first station to arriving at the FGI. This is a departure from the industrial system, where manual scheduling for ML is used which was not possible to implement in the model.

Orders in the model are only started in ML if needed from AA, creating a pull system with AA as the ordering point. After assembly in AA the products will be pushed to a predetermined FA, based on the type of product. Customer deliveries are set to 13,200 products each week during weekdays only, where products are ordered from FAFGI. Specific order data from customers are extracted from the real-world planning system and used in the model as an in-memory *SQLite* database.

$$\begin{aligned} \text{Safetylevel}_i &= \alpha_i \times \phi_i \\ \text{Orderlevel}_i &= \phi_i \times \text{Leadtime} + \text{Safetylevel}_i \end{aligned} \quad (2)$$

A proprietary software, Plant Simulation,¹ is used to construct the model, due to availability at the partner company. Calendar time—explained in the following section—is important for the shift settings and the customer deliveries. Three weeks of simulation time is used, and an additional week as warm-up time to stabilize the output. The customer orders will reach 39,600 in total during the 3 weeks where statistics are counted. Four replications are performed for every evaluation and averaged to get statistically sound results.

3.2 Optimization settings

Considering the implemented safety stock and order point system in the model, to reduce the total storage size and reduce lead time, the safety level α of each variant i , for each storage $S \in \{A, B, C, D, AA, FA\}$ will need to be minimized. The machining storages and components are represented by $m = \{A, B, C, D\}$ where $m \subseteq S$. The minimum, maximum, and step size for α_{mi} is formulated in Eq. 3.

$$\begin{aligned} \alpha_{mi} &= \begin{cases} i \in \{1, \dots, 4\} \wedge m \neq B, \\ i \in \{1, \dots, 7\} \wedge m = B \end{cases} \text{ where} \\ \alpha_{mi} &\in \{1, \dots, 20\} \text{ and } \Delta\alpha_{mi} = 1. \end{aligned} \quad (3)$$

¹ Siemens Tecnomatix Plant Simulation version 14.0.2.

Table 3 Target jobs per hour for each shift (1 ... 4) per line, blank values represent inactive shifts. Grouped by the experiment parameter $\gamma_{1...4}$

Line	γ_1				γ_2				γ_3				γ_4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
AA01	37	37	15		30	30	30		30	30	30		27	27	27	27
AA02	41	41	34		33	33	33	33	33	33	33	33	33	33	33	33
FA01	20	20							10	10	10					
FA02	17	17	15		30	30	30		20	20	20		27	27	27	27
FA03	21	21	21		21	21	21	21	21	21	21	21	21	21	21	21
FA04	20	20	13		12	12	12	12	12	12	12	12	12	12	12	12

The second optimization input controls the batch sizes β_{Ci} , for component C and each variant i . β_{Ci} , detailed in Eq. 4, controls the minimum amount to start in the line. Multiple batches of the same variant can be queued by the production system to the quantities necessary to reach the customer demand. Component C is the only component affected by setup times, thereby needing a larger batch size compared to other components. The step size, $\Delta\beta_{Ci}$, is determined by the smallest possible lot size of raw materials.

$$\beta_{Ci} \in \{32, \dots, 2048\} \text{ where } i \in \{1, \dots, 4\} \text{ and } \Delta\beta_{Ci} = 32. \quad (4)$$

The last experiment variable γ_k , where $k \in \{1, \dots, 4\}$, controls which lines are active and the paces they are producing at. By improving the synchronization in the plant, running assembly on additional time with less jobs per hour (JPH), less strain is put on the machining areas leading to a possible reduction in safety levels (Hopp and Spearman 2011). The parameter operates on 4 levels, described in Table 3. Each column 1–4, for each γ_k , represents the shifts used and the target average JPH for that shift.

The first three shifts are 40 hours each, representing the day, evening, and night shifts on weekdays. On the weekend, Shift 4 operates 24 hours consisting of weekend day time. The working hours are cumulative, e.g. γ_3 includes γ_2 and γ_1 , resulting in 120 h during the week. AA01 utilizes three shifts with an unbalanced pace in γ_1 and is balanced for either three or four shifts in $\gamma_{2...4}$. FA01 is not utilized in γ_2 and γ_4 , indicated by the lack of paces.

Only changes in AA and FA are included in the experiments, machining lines are already performing at maximum capacity, over a number of shifts. For $\gamma_{2...4}$, where assembly operates on weekends, the delivery schedule is altered removing deliveries Thursday and Friday for one customer, instead dispatching on Saturday and Sunday. Without this addition, the result would be an accumulation of finished products in FAFGI during the weekends.

The outputs are chosen based on the objectives—detailed in Eqs. (5) to (8)—average lead time in hours per product LT_S , measured on calendar time, average storage level for each storage WIP_S , number of products delivered $NumOut$, and total number of hours waiting for components for each AA WC_{lm} , Where $l \in \{1, 2\}$ represents AA01 and AA02 respectively. Due to the setup of the model, every result

Table 4 Settings for the NSGA-III algorithm used in the optimization

Parameter name	Setting
Population size	100
Reference points	84
Mutation type	Polynomial
Mutation distribution index	5
Mutation probability	0.041667
Crossover type	SBX
Crossover distribution index	20
Crossover probability	0.9

not reaching 39,600 for *maxOut* are not delivering enough products to the customer. This parameter could have been used as a constraint to the optimization, but was chosen to be included as an objective instead. The choice was made to not exclude solutions preemptively but instead increase coverage of the objective space, with the risk that the optimization algorithm prioritizes non-feasible solutions.

The first objective, Eq. 5, represents the maximum average lead time for the machining components, with the average lead time for AA and FA added. The second objective, Eq. 6, minimizes the total amount of storage in the model. Lead time and storage sizes are highly influenced by changes to α_{mi} , β_{Ci} , and paces in γ_k . The third and fourth objective—Eqs. 7 and 8 respectively—operate on one output each. The last objective Eq. 8 ensures that the solution delivers the right amount of products to the customer. Another important decision was the omission of the cost aspect. Running production on more time will add direct cost in terms of wages, but lower storage volumes and lowered lead times would lower the indirect costs. Determining the gains from the reduced indirect costs proved to be difficult, and therefore the financial aspect was removed altogether.

$$\min LT_Plant = \max_{r \in m} \{LT_r\} + LT_{AA} + LT_{FA} \quad (5)$$

$$\min LeanBuffer = \min \left\{ \sum_{s \in S} WIP_s \right\} \quad (6)$$

$$\min WaitingParts = \min \left\{ \sum_{l=1}^2 \sum_{r \in m} WC_{lr} \right\} \quad (7)$$

$$\max Out = \max \{NumOut\} \quad (8)$$

As discussed in Sect. 2.2, NSGA-III performs better for more than three objectives and was therefore chosen as the optimization algorithm. Table 4 lists the settings for the algorithm where the number of reference points were calculated using the approach from the original paper (Deb and Jain 2014; Das and Dennis 1998).

Fig. 4 The entire solution set from the optimization in a parallel coordinates chart, from Lidberg et al. (2018b)

Fig. 5 Optimization results filtered on the delivery target, *maxOut*, and colored according to the parameter γ , from Lidberg et al. (2018b)

3.3 Results from industrial case study

The optimization is set with a termination criteria of reaching 40,000 evaluations and the results from those evaluations are visualized in four sequential parallel coordinates charts in Figs. 4, 5, 6 and 7. Each figure visualizes the same data set where filters have been applied to iteratively analyze the results.

The parallel coordinates chart shows a parallel line for each evaluation, or solution, recorded in the optimization set. For this data set, 40,000 lines are shown in

Fig. 6 Only selecting solutions on the Pareto front with all previous settings applied, from Lidberg et al. (2018b)

Fig. 7 Setting limits on batch sizes β_{C1} and β_{C2} with all previous settings applied, from Lidberg et al. (2018b)

Fig. 4. The maximum and minimum values for each parameter are labelled at each respective end of the y-axis. Listed first are the four objectives, *minLeanBuffer* to *maxOut*, followed by the inputs γ to β_{C4} .

In Fig. 5 the results are filtered by $\maxOut \geq 39,600$, leaving only the evaluations delivering the desired output per week, and a coloring scheme is applied to highlight the different settings for γ_k . Yellow represents both γ_2 and γ_3 as they are considered to produce the same effect; differing in the use of one or two final assembly modules shown previously in Table 3. γ_1 is represented as red and γ_4 is represented as

Table 5 Comparison between base values, manual experiments and optimization for machining lines. Only machining lines are used to make a fair comparison to previous work

Parameter name	Base	Experiments	Optimization
Work in process	10,920	7914	6786
Leadtime (h)	49	49	18
Batch sizes	3100	3100	1248

green. This coloring scheme will be kept throughout the industrial case study results section.

Contrasting γ_1 and γ_4 , representing the maximum unlevelled flow and the maximum levelled flow respectively, γ_1 , in red, has better performance for *minLeanBuffer* and γ_4 , in green, performs better in *minLT_Plant*. Increasing the number of shifts will always improve *minLT_Plant* on the condition that a customer exists, because lead time in the model is based on calendar time.

Applying a non-dominated sorting filter retains the best i.e., non-dominated, solutions on the Pareto front, shown in Fig. 6, and nearly removes all solutions where γ_1 is used. The remaining γ_1 evaluations represent the worst performance for *minWaitingParts*, but the best for *minLeanBuffer*. The high settings for α_{B5} and α_{C3} are explained since they are low volume variants and increasing the safety level for either results in a minor increase in total storage levels, thereby not affecting the objectives sufficiently for them to be minimized. This effect can be seen as the result of Eq. 2; sufficiently small demands diminishes the importance of α_{mi} .

Finally, by limiting the values for the largest batch sizes, β_{C1} and β_{C2} , the remaining solutions have lower resulting values for the objectives *minLeanBuffer* and *minWaitingParts*. The lowest results for *minWaitingParts*, seen with other solutions, are not possible to reach with this configuration, because setup takes up a larger portion of the production time. Considering only these attributes, the setup time and batch size can be considered to be the bottleneck for further reducing storage size and lead time. In Fig. 7, the results are presented, where only values with γ_4 remain in the solution space. Following the setting for each parameter, a recipe is formed for the optimized setup of the industrial system. These settings can be applied directly to the industrial system and thereby improve it in line with the chosen objectives.

Comparing the starting conditions of the industrial system, with the results achieved for the manual experiments in Lidberg et al. (2018a), and to the optimized settings presented in this article, the improvements are summarized for ML in Table 5. Only ML are shown to allow a fair comparison to previous work. Showing only ML explains the difference between the output *LT* in Table 5 and the output *minLT_Plant* in Fig. 7 because only part of the complete plant is considered. Optimization enables the simultaneous improvement of several result parameters without manual supervision. This leads to better decision support provided to the decision maker, where she can take decisions considering a range of parameters and goals.

Fig. 8 Clustering results from k-means, generating 6 clusters. Shown with the objectives and the Shift parameter

3.4 Applying knowledge discovery and data-mining

The large number of evaluations makes the extraction of rules from the optimization data set using data-mining techniques appropriate. Which attributes are important for the high-performance solutions? Which attributes are unnecessary to improve? Which attributes are important for the transition between these regions? These are some questions we are trying to answer in this section. In contrast to the SBI approach by Dudas et al. (2014), only the Pareto-optimal solutions are used in the following analysis, 2723 evaluations.

Using k-means clustering, a partitive clustering technique, we can algorithmically assign solutions to a cluster depending on their proximity, and therefore similarity, in objective space. Using *modeFRONTIER*,² the data set is clustered by analyzing the inputs and objectives for each solution. The result of the clustering is shown in Fig. 8 and shows six clusters. When representing the clustering in a parallel coordinates chart, the values for the different parameters are shown as the mean of all included solutions as well as an area showing the confidence interval. This means that the starting points of the lines on *Shift* can be seen to be at continuous points, even though the parameter is discrete.

Analyzing the clustering shows that three clusters, 1, 3, and 5, are interesting because they deliver high numbers of *maxOut*. Clusters 1 and 5, have the worst performance for *minLeanPlantBuffer* but are comparatively better in *minWaitingParts*. Cluster 3 has better performance for *minLeanPlantBuffer* and the best performance for *minLT_Plant*, while slightly worse performance for *minWaitingParts*. To reach

² Version 2017R2, a software suite by ESTECO.

Fig. 9 Decision tree classifying clusters from the data

these clusters by decision, a decision tree classifier can be trained to evaluate the criteria needed.

The decision trees are generated with *R*, version 3.5.1, the package *rpart*, and visualized with the *rattle* package (R Core Team 2019; Therneau and Atkinson 2018; Williams 2011). The bottom nodes, called leaves, are the outcomes, and the important predictors are seen as nodes ultimately leading down to the leaves. For the graphs presented, Fig. 9 shows a classifier and Fig. 10 presents a regression tree. The outcome in Fig. 9 is categories and is thus suitable as a classifier. The bottom nodes present the probability to match either cluster depending on the rules leading there and the percentage shows the number of solutions in that leaf. For the regression tree, the outcome is numeric and the leaf shows the mean value of the outcome in that leaf. Also shown is the number of solutions matching the regression and the percentage of all solutions.

In Fig. 9, the most divisive parameter is β_{C3} , represented by the top node. It divides the data to the left side with values ≥ 48 , or to the right side with values

Fig. 10 Decision tree analyzing how to achieve minimum LT_{Plant}

below 48. Following the left side, the next divide is for the γ parameter where values over 1 results in reaching *Cluster_1* and *Cluster_4* otherwise. *Cluster_5* and *Cluster_6* are the two clusters most difficult to classify conclusively, appearing as two separate leaves.

Figure 10 shows the decision tree for the objective $minLT_{Plant}$, low scores in the leaves are beneficial and is reached exclusively by setting the *Shift* parameter to four, showing the importance of that factor. A decision tree for every objective was fitted, and the rules to reach the best solutions is shown in Table 6.

The most important factor is β , and β_{C2} in particular, for all the objectives with six out of 13 rule components being β . The manual analysis made in Sect. 3.3 is in accordance with these rules. A decision maker can follow these guidelines to reach their wanted target level for the objectives, depending on their specific preferences.

Table 6 Rules extracted for the best value of every objective from the decision trees

Objective	Rule	Mean value
minLeanBuffer	$\beta_{C2} < 688, \alpha_{D1} < 17, \beta_{C2} < 592, \alpha_{D2} \geq 2.5$	7669
minWaitingParts	$\beta_{C2} \geq 624, \gamma \in \{2, 3, 4\}, \alpha_{D1} \geq 10$	4.1h
minLT_Plant	$\gamma = 4, \alpha_{D1} < 13.5, \beta_{C2} < 1408$	37 h
maxOut	$\beta_{C2} \geq 400, \alpha_{D1} \geq 3.5, \beta_{C1} < 1296$	39,403

4 Conclusions

Analysis, and optimization, of manufacturing systems on the factory level—and not only line level—is required to identify possible improvements to the total system. Optimizing production lines, or parts of the system, leads to decisions beneficial to the individual line, which could prove unnecessary in the complete system and thereby expending resources better used elsewhere. The use of faster aggregated models, with output similar to detailed models, enables optimization of the complete system.

This article shows the potential in utilizing multi-objective optimization and knowledge extraction combined with aggregated DES modelling to allow fast insights into the operation and improvements on the factory level of an industrial system. Adding the concepts of data-mining, knowledge-extraction, and decision-support gives more insights into the optimization results, how the industrial system performs, and how the system functions. Compared to previous studies, SMO based on aggregated models gives faster delivery of results, with better accuracy for certain lines by the addition of setup times, and allows additional input and output parameters to be included.

Analyzing the correlation between the parameters of the aggregated model and the disaggregated model would give insights into the analytic capacities of the aggregation technique. Are aggregated models usable to identify bottlenecks in the production system, and which bottleneck analysis techniques can be applied? Incorporating additional sources of variation could also increase the applicability and validity of the aggregation technique.

Improving—or exchanging—the aggregation technique would enable even faster results, or allow for additional complexity in the questions put to the model. Adding costs for the different improvement suggestions would add another dimension to the decision-support given and would give decisions which are ready to be implemented when the results of the optimization study are presented.

5 Discussions and future work

Fast simulation models, as well as simplified data gathering, are needed to deliver timely decision-support through SMO. Perceived costs, both in time and inactivity, of waiting for a data based decision could result in decisions made on

intuition in the interim. The aggregation technique applied in this article can enable the fast delivery of decision-support through SMO.

While the technique is untested on other processes than discrete manufacturing, the technique is based on Little's law meaning it should be applicable on general queuing systems. The limitations of the technique in regards to the layouts possible to abstract are also not fully explored. The case study in this article has revealed situations where the model studied forces partition of the lines into multiple aggregated models. Dividing longer lines into several aggregated models based on the size of intermediate buffers resulted in better validity of the output. The exact delineations, and rules, for the application of the technique are not evaluated which could be interesting to test experimentally.

The detail lost when aggregating using the applied technique concerns both input and output. Reducing the number of inputs limits the number of possible changes to the model, and the granularity to affect the output. A detailed model would still be constrained by a specific bottleneck—varied depending on the type of system and production plan—limiting the output on a few inputs concerning the bottleneck. Although; a low number of input parameters is beneficial for reducing the time taken for data gathering. Not approached in this article is the application of the technique to conceptual modelling where the specifics of the system modelled are unknown, or detailed data is unavailable. Approximating a handful of parameters is less complex than building a complete model of a system at a conceptual stage.

Aggregating models also limits the range of output as shown in the validation chapter of this article. Large differences in processing time between variants could prohibit the use of this technique in its current form. Smaller differences can be accounted for by averaging the variant processing times, but the allowed range of processing times require additional testing by further work. The maximum *WIP*, in the validation chapter, also differ between the aggregated and detailed model. The detailed model shows a 30% larger maximum value, compared to the aggregated models, while the mean values are similar. Considering the layout differences, with a faster processing final section of the detailed model, material building up before the final section, due to setup and failures, is allowed to exit the model quicker; thereby reducing the average *WIP* but exhibiting a larger maximum value. This is one additional avenue of improvement to be made in further work.

The creation of an aggregated model does not exclude the use of the detailed, or disaggregated, model. The aggregated model is an approximation of the detailed model requiring the use of the detailed model as validation of the results produced. Optimization—where an abstracted model can evaluate quickly—should be guided by the aggregated model, but after selection of the wanted solution, the same parameters should be evaluated in the detailed model to ensure validity. This approach is conditional on the parameters used in the study, optimizing the availability of an aggregated model is not as easily transferred to the detailed model due to the issue of distributing an aggregated value across several specific availability values for each resource. Instead the aggregated model will be used to set the target values for the detailed model. The analytic capacities of an aggregated model has not been evaluated in this article but can prove to be interesting for future work.

Acknowledgements The authors wish to acknowledge Marcus Frantzén from Volvo Car Corporation, for additional guidance with the aggregation technique while writing this article. This research is funded in part by the Swedish Knowledge Foundation and by Volvo Car Corporation.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- Asmundsson J, Rardin RL, Uzsoy R (2006) Tractable nonlinear production planning models for semiconductor wafer fabrication facilities. *IEEE Trans Semicond Manuf* 19(1):95–111. <https://doi.org/10.1109/TSM.2005.863214>
- Banks J, Nelson BL, Carson JS, Nicol DM (2009) Discrete-event system simulation, 5th edn. Pearson UK, Essex
- Burghout W (2004) Hybrid microscopic-mesoscopic traffic simulation. PhD thesis, KTH, Infrastructure
- Chand S, Wagner M (2015) Evolutionary many-objective optimization: a quick-start guide. *Surv Oper Res Manag Sci* 20(2):35–42. <https://doi.org/10.1016/J.SORMS.2015.08.001>
- Chwif L, Barretto MRP, Paul RJ (2000) On simulation model complexity. In: Joines JA, Barton RR, Kang K, Fishwick PA (eds) Proceedings of the 2000 winter simulation conference, IEEE, Piscataway, New Jersey, pp 449–455. <https://doi.org/10.1109/WSC.2000.899751>
- Chwif L, Paul RJ, Barretto MRP (2006) Discrete event simulation model reduction: a causal approach. *Simul Model Pract Theory* 14(7):930–944. <https://doi.org/10.1016/J.SIMPAT.2006.05.001>
- Das I, Dennis JE (1998) Normal-boundary intersection: a new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM J Optim* 8(3):631–657. <https://doi.org/10.1137/S1052623496307510>
- Deb K (2001) Multi-objective optimization using evolutionary algorithms. Wiley, New York
- Deb K, Jain H (2014) An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach. Part I: Solving problems with box constraints. *IEEE Trans Evol Comput* 18(4):577–601. <https://doi.org/10.1109/TEVC.2013.2281535>
- Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput* 6(2):182–197. <https://doi.org/10.1109/4235.996017>
- Dudas C, Frantzén M, Ng AH (2011) A synergy of multi-objective optimization and data mining for the analysis of a flexible flow shop. *Robot Comput Integr Manuf* 27(4):687–695. <https://doi.org/10.1016/j.rcim.2010.12.005>
- Dudas C, Ng AH, Pehrsson L, Boström H (2014) Integration of data mining and multi-objective optimisation for decision support in production systems development. *Int J Comput Integr Manuf* 27(9):824–839. <https://doi.org/10.1080/0951192X.2013.834481>
- Fayyad U, Piatetsky-Shapiro G, Smyth P (1996) From data mining to knowledge discovery in databases. *AI Mag* 17(3):37. <https://doi.org/10.1609/aimag.v17i3.1230>
- Floreano D, Mattiussi C (2008) Bio-inspired artificial intelligence: theories, methods, and technologies. MIT Press, Cambridge
- Forgy E (1965) Cluster analysis of multivariate data: efficiency versus interpretability models. *Biometrics* 61(3):768–769
- Fowler JW, Rose O (2004) Grand challenges in modeling and simulation of complex manufacturing systems. *Simulation* 80(9):469–476. <https://doi.org/10.1177/0037549704044324>
- Frantz FK (2003) A taxonomy of model abstraction techniques. In: Proceedings of the 27th conference on Winter simulation—WSC’95. ACM Press, New York, New York, USA, pp 1413–1420. <https://doi.org/10.1145/224401.224834>, <http://portal.acm.org/citation.cfm?doid=224401.224834>
- Frawley WJ, Piatetsky-Shapiro G, Mathews CJ (1992) Knowledge discovery in databases: an overview. *AI Mag* 13(3):57. <https://doi.org/10.1609/AIMAG.V13I3.1011>
- Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning. *Choice Rev Online* 27(02):27-0936–27-0936. <https://doi.org/10.5860/CHOICE.27-0936>

- Hennig A, Reggelin T, Wolff D (2014) Mesoscopic simulation models for the automotive industry. In: Proceedings of the European modeling and simulation symposium, modeling & simulation Center—Laboratory of Enterprise Solutions. http://www.msc-les.org/proceedings/emss/2016/EMSS2016_309.pdf
- Holland JH (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, Cambridge
- Hopp WJ, Spearman ML (2011) Factory physics, 3rd edn. Waveland Press, Long Grove
- Jain H, Deb K (2014) An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach. Part II: Handling constraints and extending to an adaptive approach. *IEEE Trans Evol Comput* 18(4):602–622. <https://doi.org/10.1109/TEVC.2013.2281534>
- Jain S, Lechevalier D, Woo J, Shin SJ (2015) Towards a Virtual Factory Prototype. In: Proceedings of the 2015 winter simulation conference. IEEE Press, Piscataway, NJ, USA, WSC'15, pp 2207–2218. <http://dl.acm.org/citation.cfm?id=2888619.2888870>
- Koren Y (2010) The global manufacturing revolution: product–process–business integration and reconfigurable systems. Wiley, Hoboken
- Kuo Y, Yang T, Peters BA, Chang I (2007) Simulation metamodel development using uniform design and neural networks for automated material handling systems in semiconductor wafer fabrication. *Simul Model Pract Theory* 15(8):1002–1015. <https://doi.org/10.1016/j.simpat.2007.05.006>
- Lefebvre E, Armbruster D (2011) Aggregate modeling of manufacturing systems. In: Planning production and inventories in the extended enterprise, international series in operations research & management science, vol 151. Springer, Boston, MA, pp 509–536. https://doi.org/10.1007/978-1-4419-6485-4_17
- Li B, Li J, Tang K, Yao X (2015) Many-objective evolutionary algorithms. *ACM Comput Surv* 48(1):1–35. <https://doi.org/10.1145/2792984>
- Lidberg S, Pehrsson L, Frantzen M (2018a) Applying aggregated line modeling techniques to optimize real world manufacturing systems. *Procedia Manuf* 25(May):89–96. <https://doi.org/10.1016/j.promfg.2018.06.061>
- Lidberg S, Pehrsson L, Ng AH (2018b) Using aggregated discrete event simulation models and multi-objective optimization to improve real-world FACTORIES. In: Rabe M, Juan AA, Mustafee N, Skoogh A, Jain S, Johansson B (eds) 2018 Winter simulation conference (WSC). IEEE, Gothenburg, pp 2015–2024. <https://doi.org/10.1109/WSC.2018.8632337>
- Little JDC (1961) A proof for the queuing formula: $L = \lambda W$. *Oper Res* 9(3):383–387. <https://doi.org/10.1287/opre.9.3.383>
- Lloyd SP (1982) Least squares quantization in PCM. *IEEE Trans Inf Theory* 28(2):129–137. <https://doi.org/10.1109/TVT.1982.1056489>
- Negahban A, Smith JS (2014) Simulation for manufacturing system design and operation: literature review and analysis. *J Manuf Syst* 33(2):241–261. <https://doi.org/10.1016/j.jmsy.2013.12.007>
- Ng AH, Deb K, Dudas C (2009) Simulation-based Innovization for production systems improvement : An industrial case study. In: Rosén BG (ed) Proceedings of the international 3rd Swedish production symposium, SPS'09, Göteborg, Sweden, 2–3 December 2009, The Swedish Production Academy, Skövde, pp 278–286
- Ng AH, Dudas C, Nießen J, Deb K (2011) Simulation-based innovization using data mining for production systems analysis. In: Wang L, Ng AHC, Deb K (eds) Multi-objective evolutionary optimisation for product design and manufacturing. Springer London, London, pp 401–429. https://doi.org/10.1007/978-0-85729-652-8_15
- Oyarbide A, Baines TS, Kay JM, Ladbrook J (2003) Manufacturing systems modelling using system dynamics: forming a dedicated modelling tool. *J Adv Manuf Syst* 02(01):71–87. <https://doi.org/10.1142/S0219686703000228>
- Pehrsson L, Lidberg S, Frantzen M, Aslam T, Ng AH (2014) Aggregated discrete event modelling for simulation and optimisation of manufacturing systems. In: 12th international industrial simulation conference 2014, ISC 2014, EUROSIS, pp 83–90. <http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-10478>
- Pehrsson L, Frantzen M, Aslam T, Ng AH (2015) Aggregated line modeling for simulation and optimization of manufacturing systems. In: Yilmaz L, Chan WKV, Moon I, Roeder TMK, Macal C, Rossetti MD (eds) Proceedings of the 2015 winter simulation conference. IEEE, Piscataway, New Jersey, pp 3632–3643. <https://doi.org/10.1109/WSC.2015.7408522>
- R Core Team (2019) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. <https://www.R-project.org/>

- Reggelin T, Tolujew J (2008) Mesoscopic modeling and simulation of logistics systems. *Adv Logist Syst Theory Pract* 2(1):21–28
- Spearman ML, Woodruff DL, Hopp WJ (1990) CONWIP: a pull alternative to kanban. *Int J Prod Res* 28(5):879–894. <https://doi.org/10.1080/00207549008942761>
- Therneau T, Atkinson B (2018) rpart: Recursive Partitioning and Regression Trees. <https://CRAN.R-project.org/package=rpart>, r package version 4.1-13
- Toledo T, Cats O, Burghout W, Koutsopoulos HN (2010) Mesoscopic simulation for transit operations. *Transp Res Part C Emerg Technol* 18(6):896–908. <https://doi.org/10.1016/j.trc.2010.02.008>
- Williams GJ (2011) Data mining with rattle and R: the art of excavating data for knowledge discovery, 2011th edn. Use R!, Springer, Berlin
- Witten IH, Frank E, Hall MA, Pal CJ (2017) Data mining: practical machine learning tools and techniques, 4th edn. Morgan Kaufmann, Cambridge
- Zulkepli J, Eldabi T, Mustafee N (2012) Hybrid simulation for modelling large systems: an example of integrated care model. In: Laroque C, Himmelsbach J, Pasupathy R, Rose O, Uhrmacher A (eds) *Proceedings title: proceedings of the 2012 winter simulation conference (WSC)*. IEEE, Piscataway, New Jersey, pp 1–12. <https://doi.org/10.1109/WSC.2012.6465314>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Simon Lidberg is an industrial Ph.D. student at Volvo Car Corporation and the University of Skövde. Graduating from the University of Skövde in 2011, with a M.Sc. in Industrial Informatics, he started working with the simulation and optimization of production systems using DES. In 2017 he started his industrial Ph.D. studies focusing on the creation of decision support by optimizing on a factory level using aggregated data from detailed DES models.

Tehseen Aslam holds a Ph.D. in Industrial Informatics from the University of Skövde, Sweden. His research interests include simulation-based optimization using multiobjective optimization and data mining techniques for the design and analysis of supply chains as well as strategic, tactical and operations manufacturing supply chain decision making.

Leif Pehrsson is the Director of Research and Technology Development, Manufacturing Engineering Power Systems, at Volvo Car Group and an Affiliated Senior Lecturer at the University of Skövde, Sweden. He holds a Ph.D. degree in Manufacturing Systems from De Montfort University in the UK., a M.Sc. degree in Automation Engineering, and a B.Sc. degree in Mechanical Engineering. He has been working in various engineering and managerial positions within automotive industry for more than 20 years.

Amos H. C. Ng is a Professor of Production and Automation Engineering at the University of Skövde, Sweden. He holds a Ph.D. degree in Computing Sciences and Engineering. His main research interest lies in applying multi-objective optimization and data mining techniques for production systems design, analysis and improvement.