Simulation Optimization for the Stochastic Economic Lot Scheduling Problem with Sequence-Dependent Setup Times

Nils Löhndorf[1], Manuel Riel, Stefan Minner[2]
[1]Vienna University of Economics and Business, Vienna, Austria
[2]Technische Universität München, Munich, Germany
{nils.loehndorf@wu.ac.at, m@nuelriel.com, stefan.minner@tum.de}

We consider the stochastic economic lot scheduling problem (SELSP) with lost sales and random demand, where switching between products is subject to sequence-dependent setup times. We propose a solution based on simulation optimization using an iterative two-step procedure which combines global policy search with local search heuristics for the traveling salesman sequencing subproblem. To optimize the production cycle, we compare two criteria: minimizing total setup times and evenly distributing setups to obtain a more regular production cycle. Based on a numerical study, we find that a policy with a balanced production cycle leads to lower cost than other policies with unbalanced cycles.

Keywords: Inventory, Multi-Product, Lot-Sizing and Scheduling, Stochastic Demand, Sequence-Dependent Setups, Simulation Optimization

1. Introduction

The integration of lot-sizing, scheduling and safety stock planning is a major challenge in multi-product inventory optimization under stochastic demands. Customized manufacturing and the use of the same manufacturing facilities by multiple products requires investments into flexible resources which need to be managed effectively with respect to utilization and inventories. Increasing product variety poses an additional challenge. Especially in the process industry, lot-sizes and production sequences play an important role where setup times considerably decrease facility utilization (see, e.g., Kallrath (2002)). The mainstream planning approach for these problems is to decompose the integrated into several sequential planning problems, typically a deterministic lot-sizing and scheduling problem modelled as a mixed-integer program and a safety stock planning problem using stochastic models. There exist only few approaches that follow an integrated

approach, e.g., the (dynamic) capacitated lot-sizing problem under random demands (Helber et al. 2013) or the stochastic economic lot-scheduling problem (SELSP) we consider in this manuscript. Vaughan (2007) compares different scheduling and sequencing options for the stochastic economic lot scheduling problem. The literature on the SELSP is reviewed in Sox et al. (1999) and Winands et al. (2011).

One dominant assumption in many existing approaches is that setup costs and especially setup times are independent of the chosen production sequence. However, there are several practical problems where sequence-dependency plays a major role. First, there can be significant setup times between different product families (major setups), but small or negligible setup times between individual products belonging to the same family (see e.g. Karalli and Flowers (2006)). Second, setup times between products may be asymmetric, which includes the special case of one-way setup times, for instance due to different cleaning requirements or added machinery tools. Furthermore, the amount of required setup time between items of a certain family might be identical, or linearly (progressively, degressively) increasing. Overviews on available approaches for lot-sizing and sequencing problems with sequence-dependent setup costs and times are available in Allahverdi et al. (1999) and Allahverdi et al. (2008). Dobson (1992) analyzes the deterministic ELSP with sequence dependent setup times using the time-varying lot-size approach which was refined recently in Shirodkara et al. (2011). Liberopoulos et al. (2013) analyze the case of a stochastic economic lot-scheduling problem with restrictions in the production sequence.

Our model and the analysis are based on Löhndorf and Minner (2013), who analyze a similar problem under the assumption that setup times are independent of the sequence. They model the problem as a Semi-Markov Decision Problem and compare Approximate Dynamic Programming with direct policy search for fixed-cycle and base-stock policies using simulation optimization. Due to the complexity of the problem, several (meta-)heuristics combined with simulation have been proposed. Wagner and Smits (2004) suggest a local search approach. Paternina-Arboleda and Das (2005) develop a multi-agent reinforcement learning approach. Kämpf and Köchel (2006)

use a genetic algorithm-based simulation optimization approach to find the parameters of structured policies. For a more detailed literature review, we refer to Löhndorf and Minner (2013). The sequence dependence of setup times adds another dimension of complexity to the problem. Even the deterministic version of the problem, the sequence dependent economic lot-scheduling problem (SD-ELSP), requires solution of a traveling salesman problem as a subproblem. Due to the overall problem complexity, we resort to a black box approach based on simulation optimization for simultaneously finding production cycles, base-stock-levels, and production frequencies. The use of simulation increases the scope of our method, as it allows practitioners to describe a production process using a simulation model, which offers a detailed but yet user-friendly representation of the actual production process. Based on this approach, we demonstrate that straightforward solutions to the SD-ELSP or SELSP are insufficient by comparing different production policies in a numerical study. Furthermore, we investigate the influence of different setup time characteristics on the performance of the policies. We consider three different policies which all have in common that the global policy search sets the base stock levels and that production follows a fixed pre-defined production cycle.

The paper is organized as follows: In Section 2 we introduce the problem and in Section 3 we first sketch the solution methodology and then the proposed manufacturing policies to be parameterized by simulation optimization. Section 4 reports the results of a numerical study and Section 5 summarizes the main findings and future research opportunities.

2. Model

We consider the continuous time stochastic economic lot scheduling problem with $n \in \{1, 2, ..., N\}$ products. There is a single machine that can only manufacture one product at a time. If the machine state changes from one product to another, it has to be set up. This requires a deterministic, sequence-dependent setup time s_{nm} to change over from product n to product m. We further assume that the setup status is preserved over an idle period. The production for one unit of product n requires a deterministic production time p_n. During a setup or the production of a single

item, interruption is not permitted. Inventories are subject to holding cost h_n per item and unit of time and cannot exceed a maximum inventory level $\bar{y}_n$. Demand for each product n follows a compound renewal process with inter-arrival distribution F_n^A and demand size distribution F_n^D. Inter-arrival times and demand size are independent for each product and across products. As in Altiok and Shiue (1995) and Krieg and Kuhn (2002), we assume that unsatisfied customer demand is lost at cost v_n per item, but we allow for partial fulfillment of an order.

The problem can be modelled and for very small instances with few products be solved using stochastic dynamic programming, see e.g. Graves (1980) or Löhndorf and Minner (2013).

While the model can be easily extended to handle setup cost in addition to setup times, in line with Krieg and Kuhn (2002), we assume that setup cost are negligible, since "often no explicit setup cost are incurreed, and the latter are used merely to represent opportunity cost of setup times" (Federgruen and Katalan 1996).

3. Solution method

3.1. Global policy search

As in Löhndorf and Minner (2013), we propose to directly search for optimal parameters of simple policies for production control. To guide the search for the optimal parameter vector, we use the CMA-ES algorithm (Hansen and Ostermeier 2001). CMA-ES generates new candidate vectors from a multivariate normal distribution, i.e., $\mathcal{N}\big(\mu^x, \mathrm{diag}(\sigma^x)\big)$, which serves as an internal model of promising search steps. Throughout the search process, the algorithm updates the distribution's means and covariances to increase the likelihood of previously successful search steps.

Figure 1 outlines a generic formulation of the CMA-ES algorithm. Denote $\pi(\,\cdot\,;x)$ as control policy which is characterized by a (continuous) parameter vector x, and denote S^M as a sample from the state transition function of the SELSP which, for a given state S and decision $\pi(S;x)$, returns a realization of the immediate cost c, the sojourn time τ and the successor state S'. The objective of the algorithm is to search for an x that minimizes the expected average cost. The algorithm is initialized with a *guess* of the best solution, μ^x, as well as a *trust* region, σ^x, in which

(1) Input arguments: initial guess μ^x, trust region σ^x
(2) Do for $i = 1, 2, \ldots, I$
  (2.1) Get $(x^1, ..., x^K) \leftarrow G^M(K)$ from internal model
  (2.2) Do for $k = 1, 2, \ldots, K$
    (2.2.1) Do for $t = 1, 2, \ldots, T$
      (2.2.1.1) Compute $\big(c_t, \tau_t, S\big) \leftarrow S^M\big(S, \pi(S; x^k)\big)$
    (2.2.2) Compute $r^k \leftarrow \sum_{t=e+1}^{T} c_t \left(\sum_{t=1}^{T} \tau_t\right)^{-1}$
  (2.3) Update internal model $U^M\big((x^1, ..., x^K), (r^1, ..., r^K)\big)$
(3) Return best solution x^*

Figure 1 Generic policy search for production control

the solution is likely to be found. The main loop consists of three steps: (2.1) generation of a set of K candidate solutions $(x^1, ..., x^K)$ from the internal model G^M which controls the search process; (2.2) simulating the transition process for T periods and recording the average cost for each candidate policy; (2.3) updating the internal model using the sampled information.

3.2. Production policies

Denote $Y = \{Y_1, ..., Y_N\}$ as the set of base-stock levels and $Q = \{Q_1, ..., Q_J\} \in \mathbb{Q}$ as the production sequence of length J, where $\mathbb{Q}$ is defined as the power set of Q. For a given position j in sequence Q and order-up-to levels Y, the production policy is given by

$$\pi(S; x) = \begin{cases} 0 & \text{if } y_n = Y_n \ \forall \ n, \\ Q_{z(j)} & \text{otherwise,} \end{cases} \tag{1}$$

where the recursive function z is defined as

$$z(j) = \begin{cases} j & \text{if } y_k < Y_k : k = Q_j, \\ z\big(j \bmod J + 1\big) & \text{otherwise.} \end{cases} \tag{2}$$

For a given position j, the function returns the next position in the sequence for which $y_n < Y_n$, where the modulus ensures that the production cycle is repeated as soon as $j = J$.

In contrast to Löhndorf and Minner (2013), sequence-dependent setup times have to be taken into account when constructing a production sequence from policy parameters. We therefore apply an iterative two-step procedure which combines a heuristic (local) search with the global policy search to jointly optimize base-stock levels, as well as the production sequence.

3.2.1. Common cycle policy (CCP) The most simple production policy with a fixed production sequence is the *common cycle policy*, where each product is produced exactly once during a cycle. The optimal production sequence is set in advance by finding the sequence with the minimum total setup time. Since the production sequence remains constant, the global policy search merely has to set the base-stock levels. To find a production sequence which minimizes the total setup time, we resort to the Lin-Kernighan heuristic (LKH) as described in Helsgaun (2000). We have to transform the setup matrix into a distance matrix with duplicate entries, because the corresponding TSP is asymmetric (Jonker and Volgenant 1983).

The real-valued vector $x \in \mathbb{R}^N$ can be transformed into integer base-stock levels by setting $Y_n = |x_n|$. Note that the production cycle $Q = \{Q_1, ..., Q_N\}$ remains constant throughout the search, so that the global search only has to search for N policy parameters.

As in Löhndorf and Minner (2013), we propose to use a heuristic solution as initial guess based on the *common cycle solution* to the ELSP. Denote k as a safety factor and $\hat{T}$ as the common cycle time. Then, we obtain an initial policy by setting

$$Y_n = \max\left\{ \left\lfloor \mu_n \hat{T} + k\sigma_n \sqrt{\hat{T}} \right\rceil, 1 \right\} \tag{3}$$

where μ_n denotes the mean demand per unit of time and σ_n the respective standard deviation. Note that $Y_n \geq 1$ is a lower bound on the order-up-to level, since production would be zero otherwise. The common cycle time $\hat{T}$ and the safety factor k are set to

$$\hat{T} = \frac{\sum_{m=1}^{N} \sum_{n=1:n\neq m}^{N} s_{nm}}{1 - \sum_{n=1}^{N} \mu_n p_n}, \quad k = \Phi^{-1}\left(\frac{v_n}{v_n + h_n \hat{T}} \right), \tag{4}$$

with Φ^{-1} as inverse standard normal distribution. As trust region, we use $\sigma_i^x = \max\left\{ \frac{1}{2}\mu_i^x, \delta \right\}$ for all policies, with $\delta \geq 1$ to ensure exploration in case $\mu_i^x = 0$.

3.2.2. Fixed-cycle policy (FCP) The common cycle policy works well with homogeneous products. However, if some products have a higher demand than others, many unnecessary setups will be done for products with low demand, and high demand products are forced to build excessive stocks. It may therefore be better to insert setups for high-demand products into the cycle more

often. The resulting *fixed-cycle policy* (FCP) follows an idea originally proposed by Dobson (1987) for the ELSP and adapted by Federgruen and Katalan (1998) for the SELSP. The idea is to first compute the optimal production frequency for each product and then use this information to construct a production sequence.

In addition to setting $Y \in \mathbb{N}^N$ as the set of order-up-to levels, we need a set of integer frequencies $R \in \mathbb{N}^N$ from which a sequence can be constructed. The corresponding continuous parameter vector $x \in \mathbb{R}^{2N}$ can be transformed by setting $Y_n = |x_n|$ and $R_n = \lfloor |x_{N+n}| \rfloor + 1$.

Löhndorf and Minner (2013) propose a heuristic method to generate an evenly spaced production sequence from given integer frequencies. However, we have to consider that setup times are sequence-dependent and simply distributing products evenly according to their frequencies may lead to high setup times. Instead, we use the Lin-Kernighan heuristic to find a production sequence which minimizes the total setup time. For FCP, however, the LKH has to be executed during the global search, which means that the heuristic is called for each candidate parameter vector x.

As an initial guess, we use the same heuristic solution as for the common cycle policy, except that we additionally set $R_n = 1$. To avoid the scheduling of identical products in sequence, we set $s_{nn} = \infty \; \forall \; n$.

3.2.3. Balanced cycle policy (BCP) A problem with the fixed cycle policy is that it may still produce sequences with very irregular production patterns, simply because change-overs between some products are cheap, so that some products are being set up back and forth while others are not being set up at all. This again may lead to high inventory levels if products are not built frequently.

A solution to this dilemma is to introduce another optimization criterion that strikes for a balance between minimizing setup times and having a sequence with a regular production pattern. Let us define *inter-setup variability* as the standard deviation of the index difference between two setups of the same product in a given production sequence. For example, given a sequence $Q = \{1, 2, 1, 3, 2\}$, the index differences for product 1 are $3 - 1 = 2$ and $5 - 3 + 1 = 3$, so that product 1 first has to wait for two and later three setups until being scheduled for production again.

Formally, we determine inter-setup variability as follows. Denote V_n as the set of positions in the sequence where product n is being produced,

$$V_n = \{j : Q_j = n\}. \tag{5}$$

The difference δ_{nk} between the position of the k-th setup of product n and the $k-1$-th setup is given by

$$\delta_{nk} = \begin{cases} V_{nk} - V_{n,k-1} & \text{if } k > 2 \\ J - V_{n,R_n} + V_{n,1} & \text{if } k = 1\ . \end{cases} \tag{6}$$

The inter-setup variability of product n is given by the standard deviation of index difference, σ_n^Q, which is given by

$$\sigma_n^Q = \sqrt{\frac{1}{R_n}\sum_{k=1}^{R_n}\left(\delta_{nk} - \mu_n^Q\right)^2}, \quad \mu_n^Q = \frac{1}{R_n}\sum_{k=1}^{R_n}\delta_{nk}, \tag{7}$$

with μ_n^Q being as mean index difference

In contrast to the fixed-cycle policy, the balanced-cycle policy requires a multi-criteria objective function that strikes a balance between inter-setup variability and total setup times. Since total setup time and inter-setup variability have two completely different units of measurement, the objective function uses a weighted average of the two criteria. Denote $\alpha \in [0, 1]$ as the weight of total setup times. Then, the multi-criteria objective function $f : \mathbb{Q} \mapsto \mathbb{R}$ that maps the production sequence Q to a real number is given by

$$f(Q) = \alpha\left(\sum_{j=1}^{J} s_{Q_j,Q_{j+1}} + s_{Q_J,Q_1}\right) + (1-\alpha)\sum_{n=1}^{N}\sigma_n^Q. \tag{8}$$

To avoid having to hand-tune the weighting, we decode α as part of the parameter vector x so that α becomes a policy parameter which is being selected during the global search. The corresponding vector $x \in \mathbb{R}^{2N+1}$ is transformed into policy parameters by setting $Y_n = |x_n|$, $R_n = \lfloor |x_{N+n}| \rfloor + 1$, and $\alpha = \min(\max(x_{2N+1}, 0), 1)$.

Since the implementation of the LKH does not provide an interface for a custom objective function, we use a modified 2-opt heuristic to search for an optimal production sequence (see Figure 2). As input, the algorithm receives an initial production sequence, e.g., the nearest neighbor

(1) Input arguments: initial sequence Q, objective function $f : \mathbb{Q} \mapsto \mathbb{R}$
(2) While $Q' = \emptyset$ or $f(Q') < f(Q)$
  (2.1) $Q' \leftarrow Q$
  (2.2) For $i = 1$ to $J - 1$ do
    (2.2.1) For $j = J$ to $i + 1$, $j \neq i$ do
      (2.2.1.1) $Q'' \leftarrow q(Q', i, j)$
      (2.2.1.2) If $f(Q'') < f(Q')$ then $Q' \leftarrow Q''$
  (2.3) If $f(Q') < f(Q)$ then $Q \leftarrow Q'$
(3) Return Q

Figure 2 Modified 2-opt heuristic for balanced production cycles

solution, as well as function f to compare two sequences. A 2-opt move exchanges successor and predecessor of two non-adjacent products, which requires a rearrangement of the sequence of all intermediate products.

Define $m : \mathbb{N} \mapsto \{1, \ldots, J\}$ as a function that maps an integer in $\mathbb{N}$ to an integer in $\{1, \ldots, J\}$],

$$m(i, J) = \begin{cases} m(J + i) & \text{if } i \leq 0, \\ m(i - J) & \text{if } i > J, \\ i & \text{otherwise.} \end{cases} \tag{9}$$

For any number larger than J, the function is equivalent to the modulus operator, but additionally accounts for negative integers. We can now use this function to rearrange the indices of a production sequence as follows: $q : \mathbb{Q} \times \{1, \ldots, J\} \times \{1, \ldots, J\} \mapsto \mathbb{Q}$ as a function that returns a permutation of the given sequence Q for given i, j,

$$q(Q, i, j) = \begin{cases} Q' \in \mathbb{Q} : Q'_{i+k} = Q_{j-k} \ \forall \ 0 \leq k \leq j - i & \text{if } i < j, \\ Q' \in \mathbb{Q} : Q'_{m(i-k)} = Q_{m(j+k)} \ \forall \ 0 \leq k < J - i + j + 1 & \text{if } i > j. \end{cases} \tag{10}$$

The edges connecting $(i, i + 1)$ with $(j - 1, j)$ are exchanged such that the production sequence between i and j is reversed. If $i > j$, the function m ensures that any k greater than J is being mapped to the lower end of Q.

The balanced-cycle policy executes the modified 2-opt heuristic for a given parameter vector x to find a production sequence that strikes a balance between total setup times and inter-setup variability. In this way, the global search treats α as decision variable while the local search treats it as a parameter. Depending on the choice of α, the local search will obtain a different ranking of

production sequences which affects the search for an optimal sequence. Since the sequence has an immediate effect on total cost, the global search can decrease total cost by controlling α, thereby guiding the local search to find a production sequence that trades off total setup times and inter-setup variability in the chosen manner.

For an initial guess for the global search, we can use the same heuristic solution as for the fixed cycle policy. In addition, we initialize the criteria weights by setting $\alpha = \mu^x_{2N+1} = 0.5$ and $\sigma^x_{2N+1} = 0.25$. Although we found that the global search is quite robust towards the choice of the initial guess, these parameters ensure that the full range of weights is contained within the 95-percent confidence interval of the marginal distribution of candidate solutions.

4. Results

4.1. Experimental Design

To test the performance of the proposed policies, we generated instances of model parameters which are maximally different and cover a large range of values. To create an instance of the SD-SELSP, we have to specify six model parameters for each of the N products: mean demand per period, variance, lost sales cost, holding cost, sequence-dependent setup times, and production time. As in Löhndorf and Minner (2013), we aggregate these parameters by a number of design choices. We first fix the average demand and lost sales cost across all products and then express the remaining model parameters through their ratios; the variance by the coefficient of variation (CV Demand); the production time through the workload that would result from producing N products with identical demand and production rates (Load Factor); the holding cost through the ratio of holding to lost sales cost (Holding/LS Cost); the average setup time from the ratio of setup to production time (Setup/Prod Time); the product-specific setup time through a measure of setup time diversity, as well as asymmetry of the setup matrix. The ranges of the design parameters are given in Table 1.

Since the set of aggregate model parameters merely defines the averages over N product-specific parameters, we additionally define a design parameter that specifies model parameter heterogeneity

Design Parameter	Unit	Min. Value	Max. Value
Avg Mean Demand	piece	5	5
Avg Lost Sales Cost	currency	100	100
Avg Holding/LS Cost	%	0.001	0.1
Avg Setup/Prod Time	%	1	50
Avg CV Demand	%	0.5	1.5
Avg Load Factor	%	0.3	0.9
Setup Matrix Diversity	%	0.25	0.75
Setup Matrix Asymmetry	%	0	2
Heterogeneity	%	1	50

Table 1 Intervals of design parameters used in the experiment

across all products. The *heterogeneity* is given by the ratio of the largest over the smallest value in a set of model parameters. Additionally, we require all other values to be equidistant to one another. For example, if the average coefficient of variation (Avg CV Demand) is 0.5 and the heterogeneity factor is 9, then, the largest CV has to be nine times larger then the smallest CV, e.g., for five products, any permutation of the set $\{0.1, 0.5, 0.7, 0.3, 0.9\}$ would satisfy this requirement. To avoid that parameters of different sets are correlated, we apply a simple shuffle algorithm to each of the parameter sets.

Based on this experimental setup, a problem instance is given by a seven-dimensional design point. As in Löhndorf and Minner (2013), we generated $1,000$ design points for problems with $N \in \{10, 20, 30\}$ products by sampling a seven-dimensional Sobol sequence. A Sobol sequence returns a sequence of vectors with elements in $[0, 1]$ which are more evenly distributed over the unit hypercube than a sequence of vectors of pseudo-random numbers. This property turns Sobol sequences into a useful method to generate experimental designs with good space-filling properties, so that design points lie not only at the edges of the hypercube that spans the experimental area but also at its interior (Chen et al. 2006).

The novelty of this numerical study over the study presented in Löhndorf and Minner (2013) is the introduction of two properties that describe the sequence-dependency inherent in the setup matrix.

The first property *setup diversity* measures the deviation of setup times depending on the previously produced product. Setup diversity is defined as the percentage of the maximally achievable

mean absolute deviation (MAD) for a given average setup time $\bar{s}_m$,

$$\mathrm{MAD}_1(\bar{s}_m) = \frac{|N\bar{s}_m - \bar{s}_m| + (N-1)|0 - \bar{s}_m|}{N} = \frac{2(N-1)}{N}\bar{s}_m. \tag{11}$$

This definition allows us to control setup diversity by setting the MAD of each column equal to the product of $\mathrm{MAD}_1(\bar{s}_m)$ and a diversity factor. For example, if $\bar{d}$ is zero, all rows of the setup matrix are identical, i.e., s_{nm} is the same for all n for a given product m. If $\bar{d}$ is equal to one, the setup matrix is sparse, since each column has only a single element greater than zero.

The second property *matrix asymmetry* determines the difference between setup times when setting up from n to m as opposed to setting up from m to n. To control matrix asymmetry, we introduce an *asymmetry factor* $\bar{a}$ which is defined as a multiplier of the MAD of all combinations of s_{nm} and s_{mn}, i.e.,

$$\frac{2}{N(N-1)}\sum_{n=1}^{N}\sum_{m=n+1}^{N}|s_{nm} - s_{mn}|. \tag{12}$$

Since this MAD also depends on the difference in average setup times, we have to control the MAD resulting from product heterogeneity in general. We can compute this value by deriving the MAD for the problem where setup times are not sequence-dependent, i.e.,

$$\mathrm{MAD}_2(\bar{s}_1, ..., \bar{s}_N) = \frac{2}{N(N-1)}\sum_{n=1}^{N}\sum_{m=n+1}^{N}|\bar{s}_n - \bar{s}_m|. \tag{13}$$

Matrix asymmetry can now be controlled by setting the MAD of each column equal to the product of $\mathrm{MAD}_2(\bar{s}_1, ..., \bar{s}_N)$ and an asymmetry factor. If $\bar{a}$ is zero, the matrix is perfectly symmetric, i.e., $s_{nm} = s_{mn} \forall\ n, m$. If $\bar{a}$ is two, matrix asymmetry is twice as high as it would be for the sequence independent case.

A problem arises from the fact that for some combinations of $\bar{a}$, $\bar{d}$, and $\bar{s}_m$ there exists no setup matrix with the desired properties. Instead of defining a feasible set for $\bar{a}$, $\bar{d}$, and $\bar{s}_m$, we formulate a mathematical optimization problem with the objective of finding a matrix with minimum deviation from the given properties.

$$\min \sum_{n=1}^{N} u_n^2 + v^2 \tag{14}$$

$$\text{s.t.} \quad \frac{1}{N-1} \sum_{n=1:n\neq m}^{N} s_{nm} = \bar{s}_m \qquad \forall\ m \in \{1,...,N\}, \tag{15}$$

$$y_{nm} \geq s_{nm} - \bar{s}_m,\ y_{nm} \geq \bar{s}_m - s_{nm} \qquad \forall\ n,m \in \{1,...,N\} : n \neq m \tag{16}$$

$$\frac{1}{N-1} \sum_{n=1:n\neq m}^{N} y_{nm} - u_m = \bar{d}\,\text{MAD}_1(\bar{s}_m) \qquad \forall\ m \in \{1,...,N\}, \tag{17}$$

$$x_{nm} \geq s_{nm} - s_{mn},\ x_{nm} \geq s_{mn} - s_{nm} \qquad \forall\ n,m \in \{1,...,N\} : m > n \tag{18}$$

$$\frac{2}{N(N-1)} \sum_{n=1}^{N} \sum_{m=n+1}^{N} x_{nm} - v = \bar{a}\,\text{MAD}_2(\bar{s}_1,...,\bar{s}_N), \tag{19}$$

$$s_{nm} \leq s_{nk} + s_{km} \qquad \forall\ n,m,k \in \{1,...,N\} : n \neq k \neq m \tag{20}$$

$$s_{nm}, y_{nm} \geq 0, x_{nm} \geq 0, u_n \geq 0, v \geq 0, s_{nn} = \infty. \tag{21}$$

Constraints (16) and (18) are used to obtain absolute values x and y defined in (11) and (13). The absolute values are tight from below but not from above, so that the true value can be smaller but never larger. Variables u and v are slack variables if constraints (17) and (19) do not hold as equalities, in which case their deviation from equality is penalized quadratically. Constraint (20) enforces the triangular inequality so that the setup time cannot be shortened by first setting up to another product.

Note that we can only penalize a positive, but not a negative deviation without using a mixed-integer formulation. For small values of MAD_1 and MAD_2, the formulation is therefore more likely to produce a setup matrix with the best fit. For high values of MAD_1 and MAD_2, on the other hand, constraints (16) and (18) may not be binding, in which case u and v are zero. While a mixed-integer formulation may overcome this drawback, the corresponding formulation is computationally more complex and not suited for generating thousands of setup time matrices.

The transition model $S^M(\cdot)$, as well as the search algorithms and the different policies CCP, FCP, and BCP, have been implemented in Java. As compound renewable process, we simulated a *stuttering* Poisson process to generate stochastic demands. For further details, we refer to Löhndorf and Minner (2013). For all test runs we allowed the CMA-ES optimizer to generate 10,000 candidate vectors and simulate the resulting average cost.

All computations were executed in the Amazon Elastic Compute Cloud (EC2). To speed up computations, we distributed the experiment across 100 'c3.xlarge' instances using MIT's StarCluster.

4.2. Numerical Results

4.2.1. Influence of model parameters on average cost. To study the influence of the design parameters on the expected average cost, we conducted an analysis of variance (ANOVA). Since the experimental design is orthogonal, the percentage of variance in cost per product that can be explained by a design parameter is measured by the coefficient of determination (r_2). Since the total mean demand increases in the number of products and thereby the expected average cost, we use average cost per product as performance measure.

We ran separate ANOVAs on the results of all three policies, CCP, FCP, and BCP. Since we were also interested in the difference between the local search heuristics, we additionally used 2-opt in combination with CCP and FCP. The results are shown in Table 2.

	CCP/2-opt		CCP/LKH		FCP/2-opt		FCP/LKH		BCP/2-opt	
Factor	r_2	F	r_2	F	r_2	F	r_2	F	r_2	F
Number of Products	0	1.31	0	10.53	0	0.21	0	0.24	0	11.73
Avg Holding/LS Cost	0.33	3098.14	0.34	3067.92	0.3	3217.15	0.3	3143.06	0.36	4288.98
Avg Setup/Prod Time	0.19	1947.41	0.19	1822.36	0.2	2300.08	0.2	2213.05	0.2	2558.56
Avg CV Demand	0.02	202.41	0.02	201.87	0.01	111.08	0.01	111.02	0.04	538.28
Avg Load Factor	0.1	1258.15	0.09	1167.37	0.15	2095.99	0.15	2027.55	0.1	1775.31
Setup Diversity	0	0.62	0	0.57	0	1.03	0	7.22	0	0.02
Asymmetry Factor	0	3.53	0	0.01	0	1.32	0	0.35	0	32.64
Heterogenity Factor	0	20.31	0	25.18	0	22.25	0	35.54	0	4.22
Linear Model	0.69	933	0.69	899	0.73	1107	0.72	729	0.77	1316

Sample Size = 2808 x 3 policies, r_2 = coefficient of determination.

Table 2 ANOVA of the influence of design parameters on cost per product for different policies

As we can see from the last row (Linear Model), all factors together explain 69 to 77 percent of the variance in cost per product, irrespective of the policy group. Avg Holding/LS Cost, Avg Setup/Prod Time and Avg Load Factor have the highest influence on average cost per product. While a large Avg Holding/LS Cost only leads to an increase of holding cost, larger values of Avg Setup/Prod Time and Avg Load Factor lead to a higher utilization which either results in higher safety stocks or a higher number of stock-outs. The number of products does not have an influence

on average costs, which implies that the performance of the solution methods remains stable for the selected problem sizes.

Neither setup matrix diversity nor matrix asymmetry have a significant effect on the average cost of an optimized solution, which indicates that all policies are capable of compensating the effect of sequence-dependence.

4.2.2. Policy evaluation. For all policies, we recorded the mean cost per product for models with 10, 20, and 30 products, as well as low and high levels of setup time diversity and setup matrix asymmetry, using the median to split the sample. Additionally, we studied the case where setup matrix diversity was zero, in which case setup times are sequence-independent. For each instance, we divide the average cost obtained by a given policy by the average cost across all instances and all policies, which gives us the standardized cost per product.

We investigated whether a better production sequence translates into lower average cost. In Table 3, the LKH-variants of our CCP- and FCP policies clearly outperform the 2-opt variants in both cases. While CCP/2-opt only chose the best policy in about 4% of the observed cases, CCP/LKH performed best in around 10% of the test instances. Its average standardized cost is also 5% lower. Even if the CCP/LKH policy did not find the best policy, it was not as far off as CCP/2-opt. We observe similar results for the fixed cycle variants. The performance increase when using LKH over 2-opt indicates that minimizing the total setup time gets more important as the sequence-dependence of the problem increases.

We also compared the difference between common- and fixed-cycle variants. Although we expected fixed-cycle policies to outperform their common-cycle counterpart, the difference in average cost was in general not significant. This implies that an increase in setup frequency for some products does not necessarily lead to a decrease in average cost as long as the corresponding production sequence merely minimizes total setup times. In that case, due to the lower dimensionality of the search space, the common cycle policy may even become the more robust choice.

The most important result of our study is the performance increase when we include inter-setup variability as an additional optimality criterion into the search for a good production sequence. The

			CCP/2-opt		CCP/LKH		FCP/2-opt		FCP/LKH		BCP/2-opt	
Prod	Div	Asym	Mean	Frq	Mean	Frq	Mean	Frq	Mean	Frq	Mean	Frq
10	zero	all	1.41	3%	1.42	9%	1.47	5%	1.46	10%	1.29	74%
	low	low	1.01	3%	1.00	2%	0.95	7%	0.95	6%	0.75	82%
		high	1.03	2%	1.00	16%	0.97	8%	0.96	18%	0.80	56%
	high	low	1.15	2%	1.10	6%	1.11	9%	1.12	4%	0.72	78%
		high	1.07	3%	1.08	6%	1.04	5%	0.99	15%	0.76	71%
20	low	low	1.05	2%	1.02	2%	1.02	3%	0.99	7%	0.76	87%
		high	1.06	3%	0.98	15%	1.05	3%	1.02	20%	0.82	58%
	high	low	1.06	5%	1.05	5%	1.05	5%	1.05	4%	0.77	81%
		high	1.09	6%	1.04	19%	1.09	2%	1.03	18%	0.84	55%
30	low	low	1.08	4%	1.06	4%	1.05	3%	1.06	4%	0.82	85%
		high	1.11	2%	1.02	8%	1.07	1%	1.04	10%	0.80	78%
	high	low	1.15	1%	1.09	6%	1.11	6%	1.11	3%	0.79	85%
		high	1.12	2%	1.03	18%	1.12	2%	1.10	7%	0.82	70%
	Total		1.08	3%	1.04	9%	1.05	5%	1.03	10%	0.79	74%

Prod = number of products, Div = setup matrix diversity, Asym = setup marix asymmetry, Mean = standardized, mean cost per product, Frq = frequency of policy being best in class.

Table 3 Policy comparison for different setup matrix configurations

last two columns in Table 3 show the results for the balanced cycle policy. We find that on the whole, this policy performed best in 73% of all test instances. In all other cases, it was on average only 3% off. This is despite the fact that it uses the inferior 2-opt algorithm for sequence optimization. The result demonstrates that merely solving a TSP to find an optimal production sequence is often insufficient. Instead, a good production sequence strikes a balance between minimum total setup times and evenly distributed setups.

We also find that policy performance remains relatively stable across different parameter configurations, with setup matrix diversity and setup matrix asymmetry having no significant effect on the policy ranking. For the special case, where setup matrix diversity is zero, we can observe a sharp overall cost increase, which is due to the fact that there simply exists no way to decrease total setup times by finding a better cycle which directly affects system utilization und thereby cost. The existence (or non-existence) of diversity, however, does not yield into different conclusions, but rather emphasizes the additional complexity and challenge and therefore the need for methods

that explicitly take diversity and asymmetry into account.

4.2.3. Computation times. Computation times depend on the used policy as well as problem size, as can be seen from Table 4. The common-cycle policies were fastest, since the production sequence can be computed in advance. The fixed-cycle policies, on the other hand, search for a new production sequence at each step of the global search, which evidently slowed them down. Although we used a state-of-the art C implementation of the LKH and exchanged all problem information in-memory, the simple 2-opt algorithm was an order of magnitude faster than the LKH. The balanced-cycle policy was also slower than the fixed-cycle policy using 2-opt, because it tends to produce much longer sequences, but it was still around one third faster than the fixed-cycle policy using LKH.

Products	CCP/2-opt	CCP/LKH	FCP/2-opt	FCP/LKH	BCP/2-opt
10	00:11:13	00:11:09	00:12:03	01:45:19	01:40:28
20	00:17:50	00:17:43	00:23:16	07:34:09	06:08:10
30	00:24:12	00:24:01	00:47:32	22:02:59	16:17:53
Total	00:17:45	00:17:38	00:27:17	10:27:29	08:02:10

Table 4 Mean computation time per problem instance in hours, minutes and seconds.

5. Conclusion

We introduced sequence-dependent setup times into the SELSP and compared different production policies using simulation optimization. We find that a schedule with balanced setups outperforms other fixed-cycle policies. The determination of a good production sequence, which is the innovation of this manuscript, turns out to deliver reasonable average cost results so that the parameter values of setup time heterogeneity do not have a major impact on total cost.

For future research, we expect that more elaborate local search algorithms for the subproblems will yield further cost improvements. Another promising extension of the problem will be to consider multi-stage manufacturing systems and parallel machines. Additionally, an investigation of the benefits of using a stochastic model rather than a deterministic problem with expected values of demand parameters is a promising venue for further numerical studies.

References

Allahverdi, A., J.N.D. Gupta, T. Aldowaisan. 1999. A review of scheduling research involving setup considerations. *Omega* **27**(2) 219–239.

Allahverdi, A., C.T. Ng, T.C.E. Cheng, M.Y. Kovalyov. 2008. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research* **187**(3) 985–1032.

Altiok, T., G.A. Shiue. 1995. Single-stage, multi-product production/inventory systems with lost sales. *Naval Research Logistics* **42**(6) 889–913.

Chen, C.C.P., K.-L. Tsui, R.R. Barton, M. Meckesheimer. 2006. A review on design, modeling and applications of computer experiments. *IIE Transactions* **38**(4) 273–291.

Dobson, G. 1987. The economic lot-scheduling problem: achieving feasibility using time-varying lot sizes. *Operations Research* **35**(5) 764–771.

Dobson, G. 1992. The cyclic lot scheduling problem with sequence dependent setups. *Operations Research* **40**(4) 736–749.

Federgruen, A., Z. Katalan. 1996. The stochastic economic lot scheduling problem: cyclical base-stock policies with idle times. *Management Science* **42**(6) 783–796.

Federgruen, A., Z. Katalan. 1998. Determining production schedules under base-stock policies in single facility multi-item production systems. *Operations Research* **46**(6) 883–898.

Graves, S.C. 1980. The multi-product production cycling problem. *AIIE Transactions* **12**(3) 233–240.

Hansen, N., A. Ostermeier. 2001. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation* **9**(2) 159–195.

Helber, S., F. Sahling, K. Schimmelpfeng. 2013. Dynamic capacitated lot sizing with random demand and dynamic safety stocks. *OR Spectrum* **35**(1) 75–105.

Helsgaun, K. 2000. An effective implementation of the lin-kernighan traveling salesman heuristic. *European Journal of Operational Research* **126**(1) 106–130.

Jonker, R., T. Volgenant. 1983. Transforming asymmetric into symmetric traveling salesman problems. *Operations Research Letters* **2**(4) 161163.

Kallrath, J. 2002. Planning and scheduling in the process industry. *OR Spectrum* **24**(3) 219–250.

Kämpf, M., P. Köchel. 2006. Simulation-based sequencing and lot size optimization for a production-and-inventory system with multiple items. *International Journal of Production Economics* **104**(1) 191–200.

Karalli, S.M., D.A. Flowers. 2006. The multiple-family ELSP with safety stocks. *Operations Research* **54**(3) 523–531.

Krieg, G.N., H. Kuhn. 2002. A decomposition method for multi-product kanban systems with setup times and lost sales. *IIE Transactions* **34**(7) 613–625.

Liberopoulos, G., D.G. Pandelis, O. Hatzikonstantinou. 2013. The stochastic economic lot sizing problem for non-stop multi-grade production with sequence-restricted setup changeovers. *Annals of Operations Research* forthcoming.

Löhndorf, N., S. Minner. 2013. Simulation optimization for the stochastic economic lot-scheduling problem. *IIE Transactions* **45**(7) 796–810.

Paternina-Arboleda, C.D., T.K. Das. 2005. A multi-agent reinforcement learning approach to obtaining dynamic control policies for stochastic lot scheduling problem. *Simulation Modelling Practice and Theory* **13**(5) 389–406.

Shirodkara, V.A., V. Madhusudanan Pillaib, R. Sridharanb. 2011. On the feasibility of sequence-dependent economic lot scheduling problem. *International Journal of Production Research* **49**(10) 2925–2939.

Sox, C.R., P.L. Jackson, A. Bowman, J.A. Muckstadt. 1999. A review of the stochastic lot scheduling problem. *International Journal of Production Economics* **62**(3) 181 – 200.

Vaughan, T.S. 2007. Cyclical schedules vs. dynamic sequencing: Replenishment dynamics and inventory efficiency. *International Journal of Production Economics* **107**(2) 518–527.

Wagner, M., S.R. Smits. 2004. A local search algorithm for the optimization of the stochastic economic lot scheduling problem. *International Journal of Production Economics* **90**(3) 391–402.

Winands, E.M.M., I.J.B.F. Adan, G.J. van Houtum. 2011. The stochastic economic lot scheduling problem: a survey. *European Journal of Operational Research* **210**(1) 1–9.