algorithms

Article

Scheduling Algorithms for a Hybrid Flow Shop under Uncertainty

Christin Schumacher * and Peter Buchholz

Informatik 4—Modeling and Simulation, Department of Computer Science, TU Dortmund University, D-44221 Dortmund, Germany; peter.buchholz@cs.tu-dortmund.de
* Correspondence: christin.schumacher@tu-dortmund.de

Received: 30 September 2020; Accepted: 27 October 2020; Published: 31 October 2020

Abstract: In modern production systems, scheduling problems have to be solved in consideration of frequently changing demands and varying production parameters. This paper presents a approach combining forecasting and classification techniques to predict uncertainty from demands, and production data with heuristics, metaheuristics, and discrete event simulation for obtaining machine schedules. The problem is a hybrid flow shop with two stages, machine qualifications, skipping stages, and uncertainty in demands. The objective is to minimize the makespan. First, based on the available data of past orders, jobs that are prone to fluctuations just before or during the production phase are identified by clustering algorithms, and production volumes are adjusted accordingly. Furthermore, the distribution of scrap rates is estimated, and the quantiles of the resulting distribution are used to increase corresponding production volumes to prevent costly rescheduling resulting from unfulfilled demands. Second, Shortest Processing Time (SPT), tabu search, and local search algorithms are developed and applied. Third, the best performing schedules are evaluated and selected using a detailed simulation model. The proposed approach is validated on a real-world production case. The results show that the price for a very robust schedule that avoids underproduction with a high probability can significantly increase the makespan.

Keywords: scheduling; uncertainty; discrete event simulation; hybrid flow shop; scrap; local search; tabu search; machine qualifications; clustering; shortest processing time

1. Introduction

In complex production environments in the automotive industry, machine schedules have to be calculated by taking into account frequently changing customer demands and potential failures or unanticipated delays. The calculation of high quality schedules in such an environment is a challenge because the scheduling problem itself usually has significant complexity, tight time restrictions are given, and uncertainty occurs in various production parameters. Even by neglecting uncertainty, the resulting optimization problems are NP-hard and can only be solved exactly with acceptable computation times of some minutes for small and unrealistic configurations. More applicable solution methods for practical problem instances are heuristics and metaheuristics, because they require less computation time and allow one to approximate the optimal schedule with deterministic models, even for larger configurations. However, the parameters of the model have to be set according to the current situation. This means that data from the running production are used to define the actual scheduling problem; a statistical evaluation of past data allows for the determination of safety margins to compensate for scrap or unplanned demands. Before schedules, which were calculated by deterministic optimization methods, should be applied to a production with its statistical effects, they should be tested in stochastic simulation models.

The objective of this paper is to provide combined solutions to various problems at the machine scheduling level for a real application example from a supplier in the automotive industry, which describes a two-stage scheduling problem with parallel machines per stage with uncertainty in several parameters. The system layout of the application case can be found in Figure 1. The production includes 11 unrelated parallel machines in the first stage. In the second stage, there are two identical parallel machines. The restrictions and characteristics of the basic scheduling problem are machine qualifications, i.e., not every job can be produced on every machine, jobs might skip stages, and the production data show several uncertainties in demands and different production parameters, which need to be handled to provide usefully applicable schedules that perform in production with as few makespans as possible. A composition of forecasting, classification, discrete event simulation, metaheuristic, and heuristic algorithms is developed to identify demand fluctuations, including scrap rates, and to approve schedules for their use in a production environment. So, in contrast to common scheduling approaches, various problems at the machine scheduling level have to be considered in combination. First, based on the available data of past orders, risky jobs that are prone to fluctuations just before or during the production phase are identified by clustering algorithms and corresponding demands are adjusted. Furthermore, the distributions of scrap rates are estimated, and the quantiles of the resulting distribution are used to increase quantities of produced items to avoid costly losses due to unfulfilled demands. Second, we show how deterministic methods solving the described optimization problem are developed from the scheduling problem and how parameters are derived from the available data. Third, a detailed simulation model of the production has been built (see Figures 2 and 7) using the software AnyLogic 8.5 [1]. By means of simulation, it is possible to evaluate and improve schedules before they are applied in a real production environment.

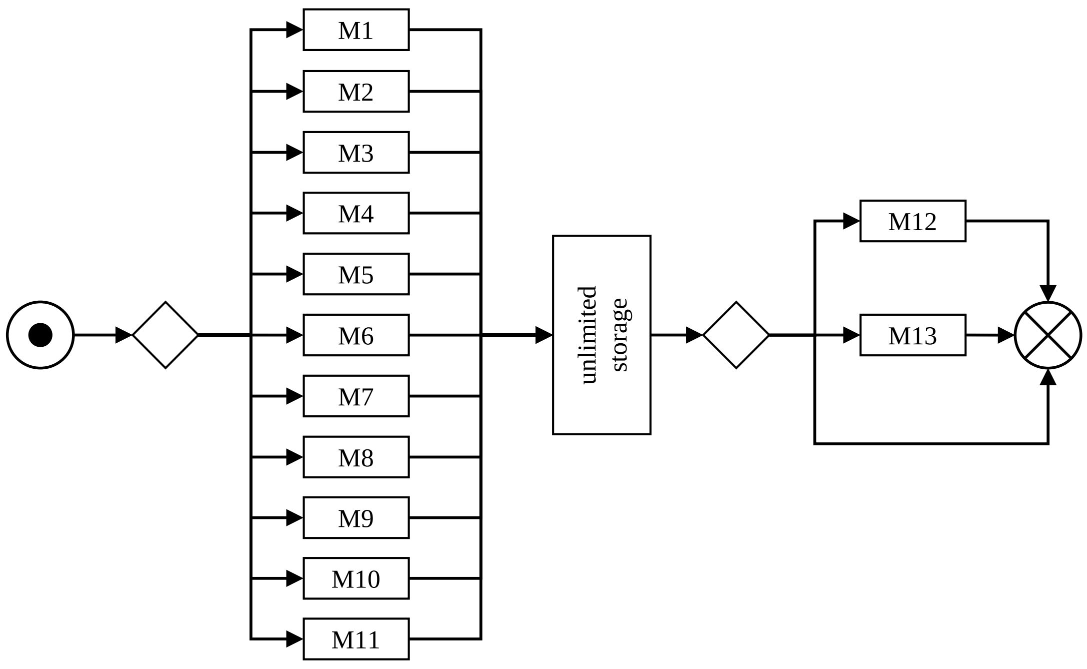

Figure 1. System layout of the hybrid flow shop in the application case.

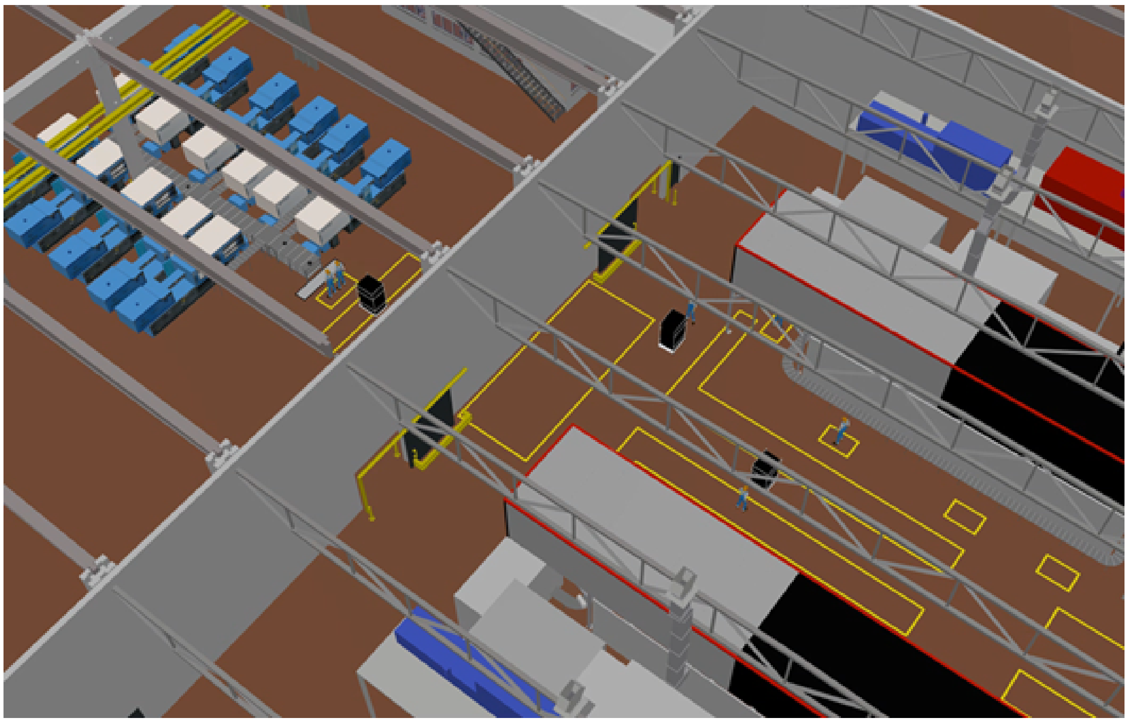

Figure 2. Production flow of the application case in AnyLogic 8.5.

The structure of the paper is as follows. In the next section, we present a detailed description of the two-stage hybrid flow shop problem with its restrictions. Related work is summarized afterwards. Section 4 introduces several metaheuristic and heuristic algorithms to compute schedules for the hybrid flow shop problem. Afterwards, in Sections 5 and 6, the available data from the production are described and methods are presented to estimate all parameters of the optimization problem and to calculate the appropriate production volume from the data. Then in Section 7, a detailed analysis of the schedules, using simulation and the structure of the simulation model, are introduced. In Section 8, results of example runs are presented, evaluating makespans and order fulfilment as key indicators. Our findings are summarized in the conclusions, which also outline some steps for future research.

2. Description of the Basic Model

The problem analyzed in this paper is a two-stage hybrid flow shop ($FH2$), in which a set of jobs $N = \{1, \ldots, n\}$ has to be completed. To produce the components, there are two production stages i, with $i \in M$ with $M = \{1, 2\}$, through which each job $j \in N$ follows the same route. At each stage i a job is processed by at most one machine l, with $l \in M_i = \{1, \ldots, m_i\}$, where $m_i \geq 1$.

As far as possible, we set the parameters according to Ruiz et al. [2] and de Siqueira et al. [3], to support a more homogeneous notation for machine scheduling studies. The following assumptions hold for our model. In brackets, descriptions of the assumptions from the notation in the paper by Graham et al. [4] are added.

1. In the first stage, all jobs are available at time zero.
2. Jobs that are to be produced in the first and last stage can only start in the second stage, after the job has been finished completely in the first stage.
3. Each machine can only handle one job at the same time.
4. Processing of a job on a machine is not interrupted, i.e., preemption is not allowed.
5. The problem contains machine qualifications (M_j), i.e., each job can only be processed on certain machines at each stage, where the set of eligible machines is E_{ij}, with $1 \leq |E_{ij}| \leq m_i$.
6. Some jobs visit only one of the stages ($skip$). The set of stages to be visited is F_j, where $1 \leq |F_j| \leq 2$.

7. Jobs do not need to be dispatched in the same order in all stages. Thus, no fix permutation is considered.
8. There are infinite buffers in front, between and at the end of the two stages.
9. Processing times p_{ilj} of one product of job j are described by independent stochastic distributions separated by machine, stage and job, i.e., machines are unrelated. Especially in the first stage, processing times differ by machine ($RM^{(1)}$), in the second stage, they are machine-independent ($PM^{(2)}$).
10. For setup times s_i, stochastic distributions over all machines and jobs per stage are available.
11. For scrap rates r_{ij}, job- and stage-related stochastic distributions are available for both stages. Scrap parts are those parts which do not pass the quality check.
12. Available parts on stock $stock_{ij}$ are included to fulfill the demand.
13. The objective is to minimize the makespan C_{max}, where C_{max} is the maximum completion time and C_{ij} defines the completion time of a job j on stage i, where $C_{max} = \max_{i \in M, j \in N} C_{ij}$.
14. Order quantities of the job's $demand_{ij}$ underlie uncertainty and may change even during the week when the job is produced, which results in varying production volumes $productionvolume_{ij}$. The needed processing time for one job is the $productionvolume_{ij} * p_{ilj}$.

According to Pinedo [5] and Ruiz and Vázquez-Rodríguez [6], this problem can be formalized in the Graham et al. [4] notation as $FH2,\ (RM^{(1)},\ PM^{(2)}) \mid M_j,\ skip \mid C_{max}$. In addition, several uncertainties in the model parameters as listed above have to be included, which are not formalized in Graham et al. [4] notation.

In addition, these other parameters of the Graham et al. [4] notation are needed in the following sections:

- S_{sd}: sequence-dependent setup times
- $avail$: block times for machines
- rm: block times for machines at the start of the production time
- lag: overlapping or gaps between the processing of jobs in successive stages
- $prec$: priority relationships between jobs, i.e., one jobs needs to begin production before another job can be started

Following Ruiz et al. [2] and to support a more homogeneous notation for machine scheduling studies, the variables x_{iljk} and x are introduced. With these variables, the schedules can be described to their fullest extent with $x = (x_{iljk})_{i \in M, l \in M_i, j,k \in N}$. The binary variable for the precedence relations x_{iljk} is defined as

$$x_{iljk} \quad := \quad \begin{cases} 1, & \text{if job } j \text{ precedes job k on machine } l \text{ at stage } i, \\ 0, & \text{otherwise.} \end{cases}$$

Furthermore, $PrioList$ is a permutation of jobs. Exemplarily, if jobs should be scheduled to released machines in the order of j_2, j_3, j_1, they are saved as $PrioList = (j_2, j_3, j_1)$.

3. Related Work

A large number of papers on scheduling in hybrid flow shops are available. Overviews can be found in Ruiz and Vázquez-Rodríguez [6], Ribas et al. [7], and Komaki et al. [8]. In the following, a selected number of studies that deal with problems related to our problem (see Section 2) are discussed.

Jabbarizadeh et al. [9] test three constructive heuristics for the problem $FHM, ((PM^{(k)})_{k=1}^{c}) \mid M_j,\ S_{sd},\ avail \mid C_{max}$. Based on Kurz and Askin [10], they assign the jobs in the processing stages $i > 1$ in the order of completion times of the previous stage. They evaluate that Shortest Processing Time (SPT) provides better results than Longest Processing Time (LPT).

A heuristic based on the algorithm of Johnson [11] gave the best results in their study. In the field of metaheuristics, a version of simulated annealing, which is a local search algorithm with the acceptance of setbacks, outperforms a genetic algorithm. In each step, the position of a randomly selected job in the first stage is changed. In the following processing steps, the jobs are dispatched according to the Earliest Completion Times (ECT) using the completion times of the previous step.

Ruiz et al. [2] present a study dealing with an enormous number of realistic components in the problem $FHM, ((RM^{(k)})_{k=1}^{c}) \mid M_j, S_{sd}, rm, lag, prec, skip \mid C_{max}$. Yet the study only compares constructive heuristics and finds that NEH, a heuristic first published for flow shops by Nawaz et al. [12], provides the best solutions.

Low et al. [13] compare for $FH2, (RM^{(1)}, 1^{(2)}) \mid M_j \mid C_{max}$ sixteen combinations of heuristics. They do not consider skipping stages. To form a production sequence for the first stage, Low et al. [13] use the following methods: random sequence, SPT using processing times of the first stage, LPT using process times of the second processing stage, and Johnson's algorithm. Once the queue is created, for each method, the jobs are assigned to the machines of the second processing stage with four different rules. They find that the modified Johnson, rule by planning the second stage according to the Earliest Completion Times (ECT) of the first stage, C_{1j} performs best.

For $FHM, ((PM^{(k)})_{k=1}^{c}) \mid S_{sd}, skip \mid C_{max}$ Naderi et al. [14] conduct job sequencing and machine assignment in the same step. All stages i are scheduled according to the Earliest Completion Times (ECT) of the stage i itself and they take the arrival times at the stages into account, if $i > 1$, which result from C_{i-1j}. In comparison to other constructive algorithms for their problem and for their test data, this constructive algorithm gives the best results. The paper also uses Iterated Local Search, which outperforms the other tested metaheuristics, like genetic algorithms, up to a number of 80 jobs. Since their problem takes sequence-dependent processing times into account, the problem in combination with applying these algorithms corresponds to unrelated machine problems.

In Burdett and Kozan [15], several constructive heuristics are compared with simulated annealing (SA) and a genetic algorithm (GA) for the computation of schedules for flow shops with non-unique jobs. Again, SA and GA outperform constructive heuristics but GA does not, in general, do better than SA. Burdett and Kozan [16] analyze flow shops with resource constraints where limited resources have to be assigned and a schedule has to be computed simultaneously. They show that an evolutionary algorithm gives good results for this class of scheduling problems. The models in the papers differ from our models in several aspects, e.g., they do not consider machine qualification, stochastic demands and scrap.

Dios et al. [17] compare 24 constructive heuristics for $FHM, ((PM^{(k)})_{k=1}^{c}) \mid skip \mid C_{max}$ and the evaluation of their experiments shows that two SPT-based and one LPT-based heuristic generate the best schedules according to C_{max}. They do not take unrelated machines into account.

Logendran et al. [18] use tabu search for $FFM, ((PM^{(k)})_{k=1}^{c}) \mid batch, skip \mid C_{max}$. Since the study works with *batch*, the algorithm contains many details that cannot be used for this work.

Kaczmarczyk et al. [19] apply tabu search to $FFM, ((PM^{(k)})_{k=1}^{c}) \mid block, skip \mid C_{max}$. For their sequence of jobs, they swap two jobs in each iteration. The positions of the two exchanged jobs are saved in the tabu list. To schedule the created sequence, the earliest available machine on every stage is chosen for the job and the stages on which the job has to be produced.

De Siqueira et al. [3] modify a variable neighbourhood search of Geiger [20] for the problem $FHM, ((RM^{(k)})_{k=1}^{c}) \mid M_j, skip \mid C_{max}$. To change the solutions, one of the six neighborhood strategies is applied randomly:

- swapping two jobs on one machine;
- swapping three jobs on one machine;
- shifting the position of a job on one machine;
- swapping of two jobs in the schedule (jobs can be scheduled on the same or different machines);
- shifting a job within a processing stage;

- relocating of a block of three consecutive jobs to a new position in a processing stage.

Thus, no study provides local search and tabu search algorithms for the problem of our paper, which is specified in Section 2. Furthermore, the above mentioned approaches are all based on a fully parametrized deterministic model. One way to optimize stochastic models is the use of sample average approximations, which are used by Almeder and Hartl [21] with variable neighborhood search to optimize a two-stage flow shop problem describing a real world production process in the metalworking industry. In contrast to our problem, the number of machines is smaller, machine qualification and setup times are not required, and the behavior of orders seems to be more homogeneous. In this case, production data are not used to determine model parameters. Instead, the sampling averaging of simulation results is used to determine the parameters of the optimization problem, which are analyzed with similar methods to the ones that we apply, but tabu lists do not seem to be used.

Burdett and Kozan [22] studied buffering in the area of machine scheduling for a parallel machine environment with one stage. They add idle times to the schedule, which depend on the variance of the processing times of the jobs. Uncertainties regarding demand are taken into account by limiting the number of jobs that can be produced in a given production period or by completely blocking resources for unexpected demand. Their study differs from our study since the variance of the processing times in our application case plays a minor role, whereas scrap rates have high influence on the demand. In addition, we can use significant historical data of demand developments to adjust the demand according to the articles through clustering.

Our approach is also related to approaches that combine simulation and optimization [23]. Regarding the different possibilities in this area to combine optimization and simulation, our study is allocated in the category of first computing a schedule for a deterministic model, which is evaluated afterwards in the detailed simulation model. In a further step, schedules can be enhanced based on the simulation results. Juan et al. [24] introduced this technique as simheuristics.

In application scenarios, the model has to be built, i.e., parameters have to be estimated and demands have to be forecasted. For parameter estimation and modeling from available data, standard methods of input modeling, as summarized, for example, in Law [25], may be applied. Classification of products according to their future demands is more demanding. Murray et al. [26] have examined an application case of order classification in production planning, but they use k-means clustering in order to group customers into segments. The main difference to the problem analyzed in this paper is in the position in the supply chain. Customers are the first layer in a supply chain because they are causing the demand for orders. The demand becomes more distorted and volatile when customers' order quantities are planned through the different stages of the supply chain [27]. We consider a problem which is more at the end of a supply chain, so the so-called bullwhip effect can be intense. To forecast demands, regression or time series can be applied. There is no one method with an optimal parameter set that is best to predict demand in different settings; instead, algorithms are developed to select good parameters Kück and Scholz-Reiter [28] or to even select the optimal prediction method together with the parameters, as carried out by Scholz-Reiter et al. [29]. According to our results and in order to generate a robust schedule, it is sufficient to oversupply the demand for high-risk jobs, which are identified by clustering algorithms.

Although we do not develop new optimization techniques, the originality of our study is that, to the best of our knowledge, it combines the statistical analysis of production data with heuristic and metaheuristic optimization methods to the specific problem of this paper and the subsequent detailed simulation of a real production problem. The experiments give insights into the behavior of different local search metaheuristics and indicate as expected that local search metaheuristics are able to improve the makespan significantly compared to schedules resulting from simple heuristics like SPT. Moreover, local search and tabu search metaheuristics to the best of our knowledge have not been adapted to the specific problem of this paper in the literature before.

4. Computation of Schedules

Taking the findings of Section 3 into account, we choose to apply SPT, local searches and tabu searches, and ECT schedules the jobs for the basic model described in Section 2. We also apply algorithms for the restriction of machine qualifications, skipping stages, and unrelated machines. At first, the first stage is scheduled with one of the algorithms, which then is combined with ECT in each case to schedule the second stage. Thus, we obtain Algorithms 1–4, which were already presented in our conference paper Schumacher et al. [30].

To create an initial solution, we generate feasible schedules with SPT (Algorithm 1), which uses ECT (Algorithm 2). With Algorithm 1, jobs are ordered according to their increasing average processing times in stage 1 and after that they are scheduled successively to stage 1 on the machines, which becomes available. When the scheduling of all jobs to stage 1 is finished, stage 2 is scheduled with Algorithm 2 according the job completion times of stage 1.

Algorithm 1: Shortest Processing Time, SPT

```
1.  Order jobs $j$ according their increasing average processing times in stage 1
    ($productionvolume_{1j} \cdot p_{1j}$) and save the queue in $PrioList$.
2.  Whenever a machine in stage 1 becomes available, select next unscheduled job
    $j$ of $PrioList$ that is qualified for the given machine and schedule $j$ on the
    available machine.
3.  Execute Algorithm 2 (ECT).
4.  **return** schedule $x$ with $C_{max}(x)$.
```

Algorithm 2: Earliest Completion Time, ECT

```
1.  Order jobs on stage 2 according to their completion times $C_{1j}$. If a job is not processed
    on stage 1, set $C_{1j} := 0$. Of course, if a job should not be processed on stage 2, the job is
    not in the sequence for stage 2. If two jobs have the same completion time $C_{1j}$, order
    these jobs alphanumerically. Save queue in $PrioList$.
2.  Whenever a machine in stage 2 becomes available, select next unscheduled job $j$
    of $PrioList$ that is qualified for the given machine and schedule $j$ on the available machine.
```

Based on this initial solution, six algorithms of Algorithms 3 and 4 optimize the solution. Both can be computed with one of the variants "shift" or "swap". With "shift", one randomly selected job in each iteration is positioned elsewhere in the existing schedule. In contrast, "swap" exchanges the positions of two randomly selected jobs. After one of these moves and the consideration of the eligibility restrictions for machines and stages, the resulting new schedule is compared with the old schedule and checked for improvement by computing and comparing makespans. Random Descent generates a new solution out of the existing one and if this solution performs better, regarding C_{max}, the new solution is the starting point for the new testing of solutions. In contrast, Steepest Descent first tests for one job j the solutions in the neighborhood of the current solution, which are created by shifting or swapping that job to all the other machines that are eligible for that job. Furthermore, Algorithm 4 can be executed with or without tabu list ($tabu \in \{true, false\}$). If tabu list is used, the algorithms avoid testing a solution x again which has already been tested and is currently part of the tabu list. So, the tabu list contains elements x, which have been tested before. If there is no improvement in the makespan within a predefined amount of iterations, Algorithms 3 and 4 terminate. By choosing all possible combinations of $method \in \{shift, swap\}$ and $tabu \in \{true, false\}$ in Algorithm 4, we have four variants of that algorithm. By computing the two possibilities $method \in \{shift, swap\}$

in Algorithm 3, we have two further algorithms to evaluate. So, in total, with SPT, we get seven different optimization algorithms for the basic model of Section 2.

Algorithm 3: Local Search—Random Descent

```
1.  Given a feasible initial solution $x$ with makespan $C_{max}(x)$.
    Choose parameter $iterations > 0$ and set $termination := iterations$.
    Choose parameter $method \in \{shift, swap\}$.
2.  while $termination \neq 0$
    (a) Duplicate $x_n := x$.
    (b) Randomly choose a job $j$ on stage $i = 1$. For this job $\exists! x_{ilkj} = 1$. According
        to $x_{ilkj} = 1$ define $l$ and $k$.
    (c) Randomly choose a machine $l_n \in E_{ij}$.
    (d) if $method = shift$
          i.  Shift job $j$ to machine $l_n$ and choose randomly a position to insert job $j$
              on this machine $l_n$.
    (e) if $method = swap$
          i.  Choose randomly a job $j_s$ on selected machine $l_n$. For this job $\exists! x_{il_nk_nj} = 1$.
              According to $x_{il_nk_nj} = 1$ define $k_n$.
          ii. Exchange positions of selected jobs $j$ and $j_s$ with setting $x_{ilkj} = 0$, $x_{il_nk_nj_s} = 0$,
              $x_{ilkj_s} = 1$, and $x_{il_njk_n} = 1$.
    (f) Save solution in $x_n$, execute Algorithm 2 (ECT) and compute makespan $C_{max}(x_n)$.
    (g) if $C_{max}(x_n) < C_{max}(x)$
              $x := x_n$, $C_{max}(x) := C_{max}(x_n)$.
              $termination := iterations$.
        else
            $termination := termination - 1$.
3.  return $x$ with $C_{max}(x)$.
```

Algorithm 4: Tabu and Local Search—Steepest Descent

```
1.  Given a feasible initial solution $x$ with makespan $C_{max}(x)$.
    Choose parameter $iterations$ and set $termination := iterations$.
    Choose parameter $method \in \{shift, swap\}$.
    Choose parameter $tabu \in \{true, false\}$.
2.  if $tabu = true$
        Initialize tabu set $T := \{x\}$.
        Choose parameter $t > 0$.
    else
        Initialize tabu set $T := \{\}$.
3.  repeat $termination \neq 0$
    (a) Duplicate $x_n := x$.
    (b) Initialize set $S := \{\}$.
        for all jobs $j$ on stage 1.
            For the selected job $\exists! x_{ilkj} = 1$. According to $x_{ilkj} = 1$ define $l$ and $k$.
            for all $l_n \in E_{ij}$
                if $method = shift$
                    Shift job $j$ to machine $l_n$ and choose random a position to
                    insert job $j$ on this machine $l_n$.
                    Save solution in $x_n$, execute Algorithm 2 (ECT) and compute
                    makespan $C_{max}(x_n)$.
                    if $x_n \notin T$
                      $S = S \cup \{x_n\}$.
                if $method = swap$
                    for all $j_s$ on the selected machine $l_n$
                      According to $x_{il_nk_nj_s} = 1$ define $k_n$.
                      Exchange positions of jobs $j$ and $j_s$ with $x_{ilkj} = x_{il_nk_nj_s} = 0$, $x_{ilkj_s} = x_{il_nk_nj} = 1$.
                      Execute Algorithm 2 (ECT).
                      Save solution in $x_n$ and compute makespan $C_{max}(x_n)$.
                      if $x_n \notin T$
                        $S = S \cup \{x_n\}$.
                if $tabu = true$
                    $T = T \cup S$.
                    if $|T| > t$
                        Delete the $|T| - t$ elements from $T$, which are
                        added earliest.
            if $S = \{\}$
                Set $termination := 0$.
            for all schedules $x_n \in S$.
                if $C_{max}(x_n) < C_{max}(x)$
                    $x := x_n$, $C_{max}(x) := C_{max}(x_n)$.
                    $termination := iterations$.
                else
                    $termination := termination - 1$.
4.  return $x$ with $C_{max}(x)$.
```

In the following, the algorithms of [30] are presented.

5. Parameter Uncertainty

All algorithms presented in the previous section are assumed to have full access to the information about the problem. However, this is rarely found in practical scenarios. Usually, information about a system includes uncertainty and some parameters like processing times underlie statistical fluctuation. Some of the parameters can be described by statistical models like distributions or stochastic processes, whereas for other parameters like machine breakdowns only very little information is available.

If uncertainty is modeled by stochastic distributions, a stochastic optimization problem can be formulated; see Van Hentenryck and Bent [31]. The complexity of solving stochastic optimization problems is higher than the complexity of solving the related deterministic models. For realistic stochastic hybrid flow shop problems, even the analysis of a single configuration cannot be evaluated analytically with exact methods. Instead, for optimization of such models, stochastic discrete event simulation has to be coupled with metaheuristic optimization methods; see Juan et al. [32]. Stochastic simulation needs a lot of computation time for function evaluation and metaheuristic optimization methods often need numerous function evaluations. Therefore, the computation of nearly optimal schedules and the evaluation with simulation in every iteration with a random initial schedule can exceed available computation time in production planning of mostly only a few minutes. In order to decrease the computation time, it is more efficient to compare different near-optimal schedules from deterministic optimization models using a detailed simulation model. Furthermore, any model approximates the real system, so that the best solution for the simulation model or deterministic model is not automatically the best schedule for the real-world system. Finding a high-quality schedule which is robust against small changes in the parameters or the description of the random variables is more important.

In production systems, a large amount of data from business information systems are often available. These data can be utilized to model uncertainty. We have to differentiate between internal parameters of the production system on the one hand, like processing times, setup times, scrap rates, and the availability of machines and external parameters, which are mainly described by the varying demand, on the other. We begin with the internal parameters that, to some extent, are under the control of the company and can be measured in the production system. This, of course, does not imply that uncertainty can be deleted from the system but it is often possible to apply standard methods from input modeling to generate a distribution that appropriately models the field data or to use the cleaned data as an empirical distribution.

The availability and quality of production data vary considerably. In Table 1, we analyzed scenarios where data are available in different levels of detail, and show how these data can be used in deterministic and stochastic models. Another issue to consider are effects of outliers. Due to their high values, it is more realistic for models to use the median instead of the mean for processing times and setup times to compute more realistic schedules by using the deterministic optimization model. In contrast, for scrap rates, it is recommended to use the mean value or some quantile to avoid underproduction. If the needed data are available, the highest category of detail of Table 1 should be used for the models. It is possible that job-related and machine-related distributions, even for machines of the same type, like in our application case, differ significantly. Differences result from detailed machine conditions even if machines are nearly identical. For example, in Figure 3, the processing times of similar jobs on different machines of one type are presented.

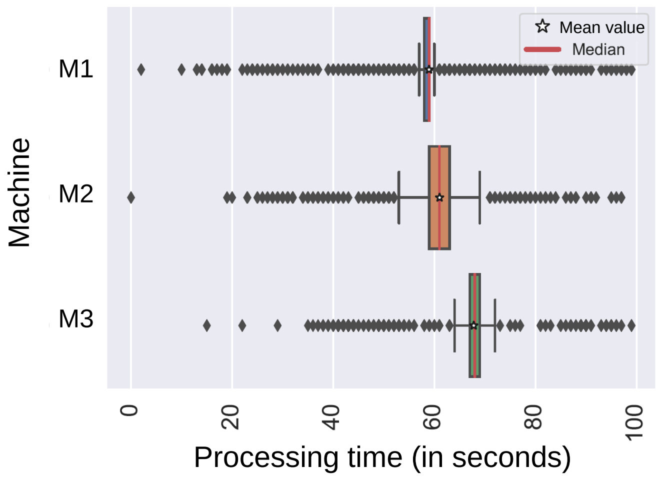

Figure 3. Boxplot for processing times for one exemplar of similar jobs j on different machines at stage 1 of the same machine type.

Table 1. Data concept for scheduling parameters in optimization and simulation.

	Processing Times p_{ilj} and Setup Times s_i		**Scrap Rate r_{ij}**	
Scenario	**Optimization**	**Simulation**	**Optimization**	**Simulation**
data on dedicated machine available	median per job and machine	random value from distribution per job and machine	mean value or value from quantile per job and machine	random value from distribution per job over all machines
no data on dedicated machines available for the job, but job data at the stage available	median per job over all machines	random value from distribution per job over all machines	mean value or value from quantile per job over all machines	random value from distribution per job over all machines
no job-related and machine-related data available, but estimates for job data are available from external sources (e.g., expert knowledge)	estimated data for the job at the stage	estimated data for the job at the stage	-	-
no job-related and machine-related data available, but data for all jobs at the stage available	median over all machines for all jobs	random value from distribution over all machines for all jobs	mean value or value from quantile over all machines for all jobs	random value from distribution over all machines for all jobs

External uncertainty, which is part of the problem in this paper, caused by varying demands, is out of the control of the producer. Future demands can be predicted based on historical data. In particular, at the first levels of a longer supply chain, uncertainty in demands grow in volatility and amount throughout the levels [27]. Figure 4 gives an example of demands over twelve weeks of one specific job and the same production date from a supplier in the automotive industry. The value in the last week in each time series shows the demand that has to be finally supplied to the customer. The other observations illustrate the development of the previously mentioned amount from the customer for the same date.

In the example of Figure 4, the demand grows in the last week considerably. Thus, if a schedule has been computed with the data of one week in advance, the production will produce less than is needed. Moreover, during the planned week itself, changes occur for high-risk jobs. Fluctuation in demands is, of course, job specific and it is not necessary or possible to model every job in detail. Hence, we first perform a cluster analysis based on historical data to distinguish between low, medium and high-risk jobs. A high-risk job changes its demand often and significantly in the weeks before and in the week of production, whereas for low-risk jobs, ordered quantities and final demand are very similar. For high-risk jobs, we use over-provisioning to avoid situations where the demand cannot be satisfied. The method is specified in the next section.

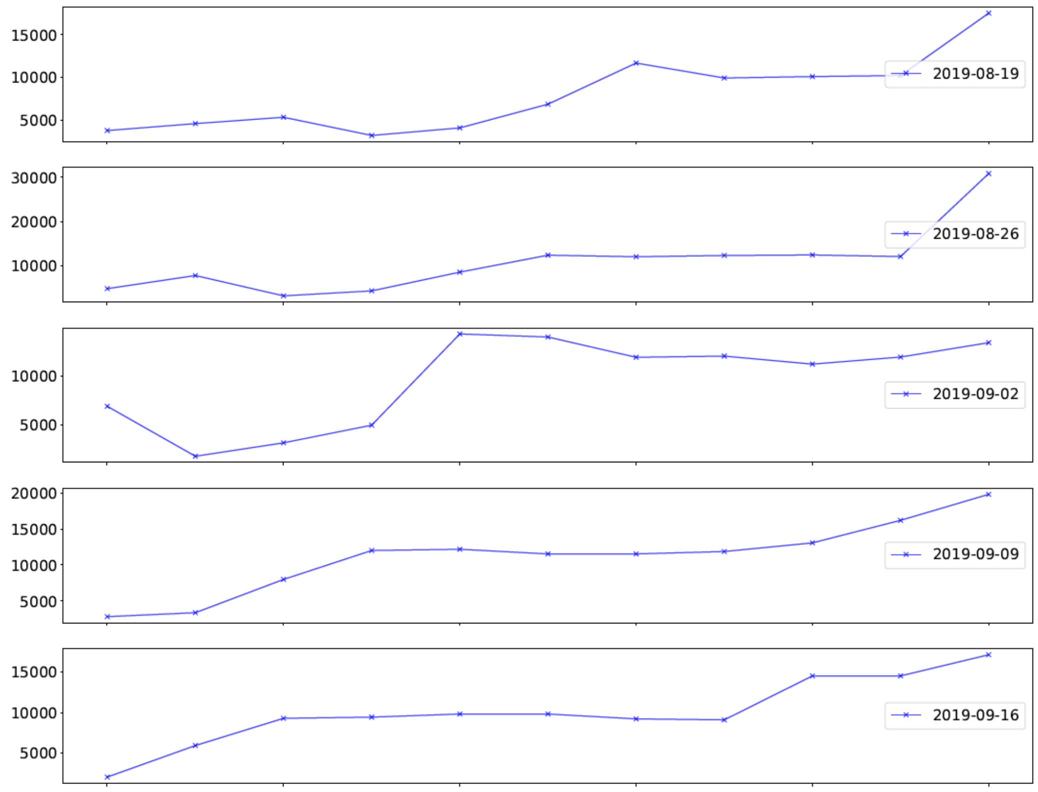

Figure 4. Time series of demands of a high-risk job.

6. Clustering and Forecasting in the Application Case

In order to allocate the appropriate production volume ($productionvolume_{ij}$) per job and stage and to meet the final customer demand volume, scrap and customer ordering behavior have to be considered. Therefore, the currently known demand per job and stage ($demand_{ij}$) as well as historical data of scrap and demand are used. The procedure of including these factors is described in the following and finally results in Equation (1).

The clustering procedure was first mentioned in Schumacher et al. [30]. For clustering demand data and identifying high-risk jobs (see Figure 4), k-means++ is applied with three parameters and the result is shown in Figure 5. For k-means++, Arthur and Vassilvitskii [33] theoretically prove and show in their results the quality of the algorithm against k-means algorithms for clustering. The three parameters for clustering are namely:

- the probability that the demand of a job grows by 500 or more parts in the last week based on historical data (emp_prob);
- the average increase in demand per job (only counted if the demands are increased) ($mean\_pos\_diff$);
- and the variance of the variation coefficient for the demands of the last three observations of the previously ordered amount. ($var\_vark\_last\_3$).

Following the elbow method, see Leskovec et al. [34], k-means should be applied for our application case with four clusters. After analyzing the data, we computed the four clusters as shown in Figure 5 and defined the jobs of cluster 3 and 4 (marked in light red in Figure 5) as high-risk jobs. The demand for high-risk jobs is multiplied with job-related safety factors, namely $SF2_{ij}$, where $1 \geq SF2_{ij} \geq 2$. The safety factor is used to keep the slot for the higher demand in the schedule reserved. Instead of only planning with the $\frac{demand_{ij}-stock_{ij}}{(1-r_{ij})}$ for high-risk-articles, the production volume for risky jobs is extended in this first step to $\frac{demand_{ij}-stock_{ij}}{(1-r_{ij})} \cdot SF2_{ij}$, where $SF2_{ij} \geq 1$ for high-risk jobs j.

The demands of non high-risk jobs are multiplied with $SF2_{ij} = 1$. These considerations are also finally included in Figure 6.

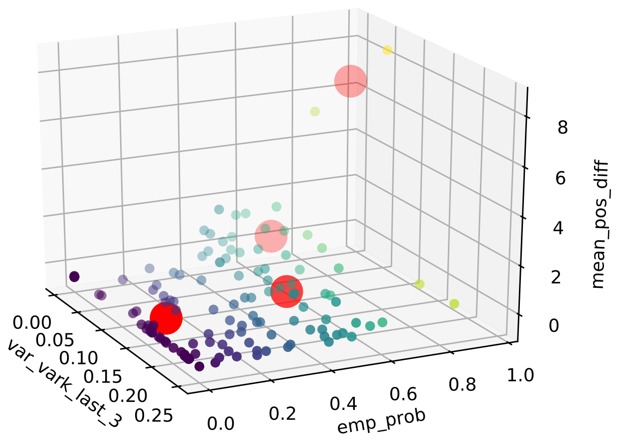

Figure 5. Clusters for identification of high-risk jobs (calculated with the method of [30]).

The same procedure of reserving slots for required higher amounts in the schedules can also be applied to other non high-risk job demands, e.g., if products are sensitive about producing less than the required demand. So, for these jobs, this method for overproduction compared to $demand_{ij}$ can be applied. The demands can be multiplied with another safety factor, namely $SF1$. This bears the possibility to include the knowledge of experienced production planners, e.g., about sensitive demands. A production planner may decide which value to use for the safety factor. At this point, it is important to note that an earlier completion, which can be the cause of slots being reserved for too long in the schedule, is easy to handle with our algorithms presented in Section 2. Such a schedule causes, at most, only a slightly extended storage cost. In contrast, an unexpected longer slot for jobs would result in drastic delays in the schedule and elaborate rescheduling. With this method, changes in the demand and production quantities can be considered in optimization and simulation models for scheduling.

To choose the appropriate scrap rate per job and stage for our application case, different quantile values of the available empirical probability distribution per job and stage are considered. Figure 6 shows a histogram for the scrap rate of an exemplary job. For every job, we use the same quantile value. Afterwards, per job, the scrap rates r_{ij} are computed as a function of the quantile value and the probability distribution.

In practice, it is not realistic that the production volume $productionvolume_{ij}$ is completely produced without failure. So, for every job, we have to increase the production volume to finally get the demand that is to be produced. All analyzed factors influencing the demand in the production process in our application case result in the following formula to calculate the appropriate production quantity $productionvolume_{ij}$:

$$productionvolume_{ij} = \frac{demand_{ij} - stock_{ij}}{(1 - r_{ij})} \cdot SF1_{ij} \cdot SF2_{ij} \qquad (1)$$

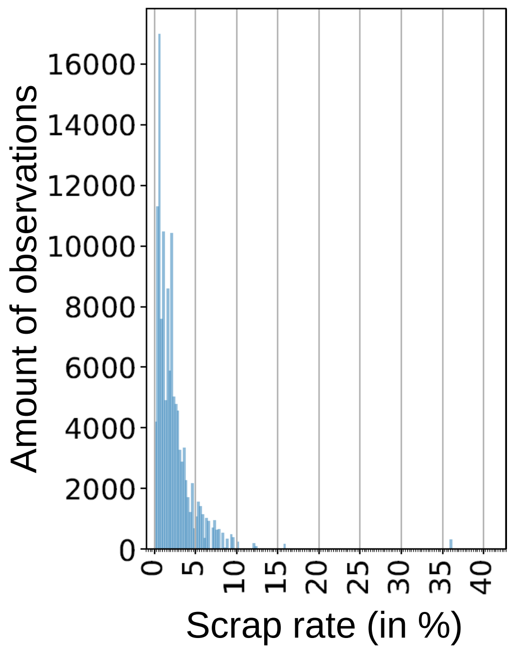

Figure 6. Scrap rate histogram for an exemplary product.

7. Scheduling under Uncertainty

Figure 7 shows the simulation architecture for our application case, which is built in Anylogic 8.5. In contrast to the deterministic optimization model, the discrete event simulation model uses random values, as mentioned in Table 1, for every produced part. Beyond that and the aspects that have already been considered in the optimization model, the following specific characteristics of production are included in the simulation model:

- In the first stage, there are two cranes and one crane needs to be used for every setup process of a machine. The first crane can only reach a defined group of machines in the first stage and the second crane can only reach the other machines. Therefore, the machines in the first stage in the model are divided into two different groups and only two machines can be set up at the same time if they do not belong to the same group, i.e., if they do not need the same crane for the setup process.
- Produced products, which are transported by conveyors to a quality check station in the first stage, can cumber each other. This can cause time losses in the production process.
- If there are not enough parts available from the first stage for producing the amount of the second stage, the production volume of the second stage has to be reduced.
- To transport the products, which were produced at the first stage, to the second stage, they are carried on small load carriers, where a certain number of produced parts fit in one small load carrier and a certain number of small load carriers fits on one trolley, which transports the products to the second stage. Both the packing and the logistics process flow between the two stages are considered.
- Detailed modeling of the connection of logistics and production between the first and second stages.
- Shift times: One of the two machines at the second stage does not produce at night and both machines at stage two do not produce on weekends.
- There are four quality checking stations, where scrap parts are identified. All stations are modeled in the simulation model with separate job-related distributions for scrap parts.

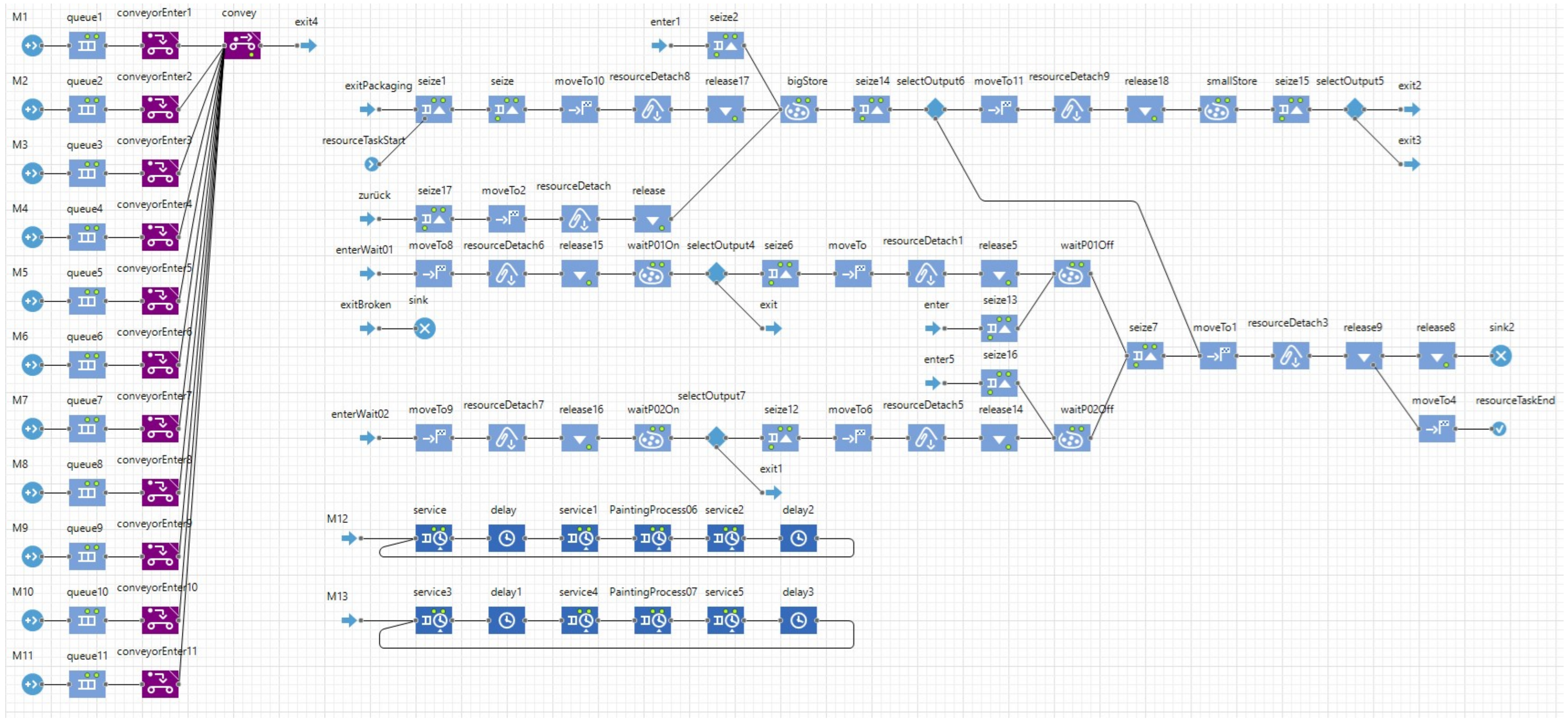

Figure 7. Structure of simulation model for the application case of the paper in AnyLogic 8.5.

Since the machine qualifications are already considered in the algorithms of Section 4, an additional validation in the simulation model is not needed.

For an almost realistic analysis of a schedule, simulation has to be used. However, a simulation run under realistic conditions is costly and should not be applied to evaluate clearly sub-optimal schedules that appear during optimization. Thus, schedules are first generated by the algorithms of Section 4 from the deterministic model and the best solutions from the deterministic model are subsequently analyzed with simulation. In a simulative analysis, several replications are performed to achieve confidence intervals of a predefined width for C_{max}. The variance of the estimator for C_{max} is also an indicator of the robustness of the schedule; a small variance indicates a predictable behavior, which is important in an industrial environment. Uncertainties resulting from scrap (see Section 5) are analyzed by counting the number of runs where the required demand is produced and the runs that fail to produce enough. The latter situation has to be avoided whenever possible in the given case study.

8. Results

To select one of the schedules created by the seven algorithms, described in Section 4, that should be applied in the application case and to select the most preferable algorithm for the application case over a defined time horizon, we derive all the algorithms with the historical demands and process data of the production. On the one hand, we evaluate the C_{max} values for the algorithms. On the other hand, we test different values for the quantile to reduce and evaluate the risk of producing less demands than the customer order (Sections 5 and 6) and simulate the calculated schedules with our simulation model, described in Section 7. For the evaluation of different security factor levels, e.g., for SF_{ij}, we refer to Schumacher et al. [30].

The specific characteristics of the evaluation data for the application case are as follows:

1. Every week, around 60 jobs are processed in the two-stage system.
2. In the first stage, there are 11 unrelated parallel machines. In the second stage, there are two identical parallel machines.
3. In both stages, stochastic distributions for p_{ilj} are available. The median at stage 1 is about 80 s per job and at stage 2 it is 4 s per job.
4. In stage 1, for setup times s_i, the stochastic distribution is available for all machines and products. The median at stage 1 is about 100 min. The median at stage 2 is about 4 min and the given data are deterministic because the second stage is a conveyor belt production stage with a constant flow rate.
5. For the scrap rate r_{ij}, job-related stochastic distributions are available for both stages.

We have applied the optimization algorithms on a computer with an Intel® Core™ i7-6920 2.9 GHz and 32 GB RAM and we have run the simulation on a Intel® Xeon™ E5-2699 v4 2.2 GHz and 64 GB RAM and an Intel® Core™ i5-4670 3.4 GHz and 16 GB RAM.

The first step for scheduling is to run the seven optimization algorithms that include deterministic processing times and setup times. Therefore, we compare the makespan resulting from the algorithms for six selected calendar weeks in the past. We run the six metaheuristics with the parameter value $iterations = 1000$ and their initial solution is given by SPT. After running the optimizations, for each week and each quantile value, each algorithm provides an objective value so that, for each algorithm, we get six data points per quantile. The analysis is illustrated in Figures 8–13.

In Schumacher et al. [30], we also analyzed the variation in objective function values caused by stochastic components in the metaheuristics for an exemplary week. Steepest Descent and tabu search algorithms provide the same objective value without variation over different runs. In contrast, the random descent algorithms provide more varying schedules. The majority of their values differ within about a quarter of a day, which means that the influence of the stochastic is relatively small in all of the developed metaheuristics of this paper.

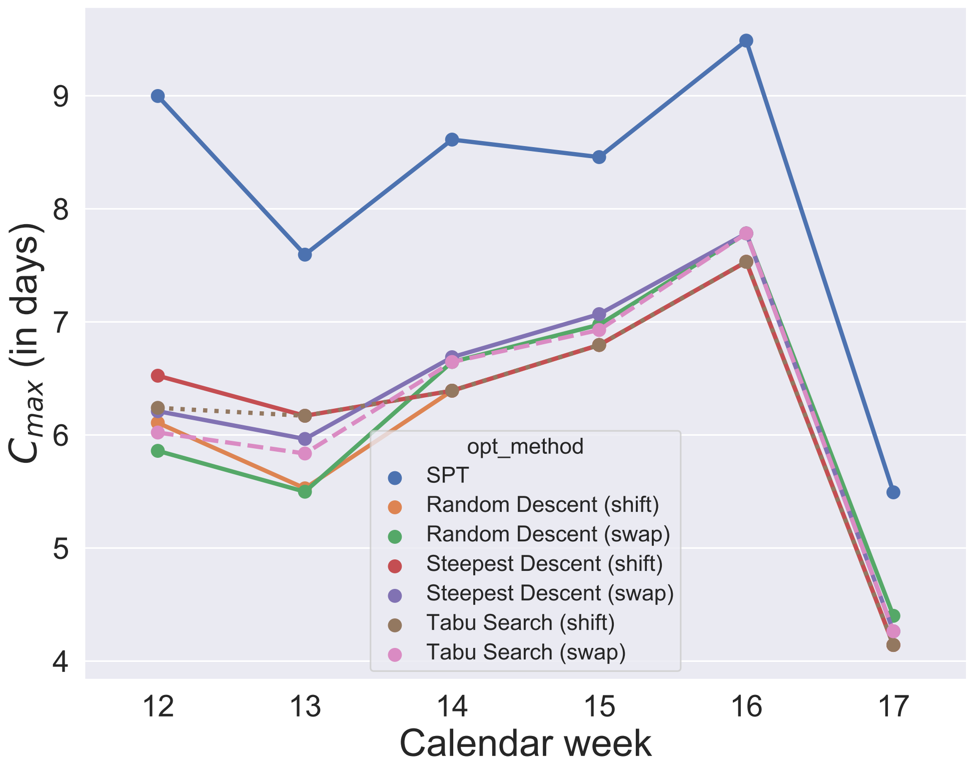

Figure 8. Optimization results for quantile 60%.

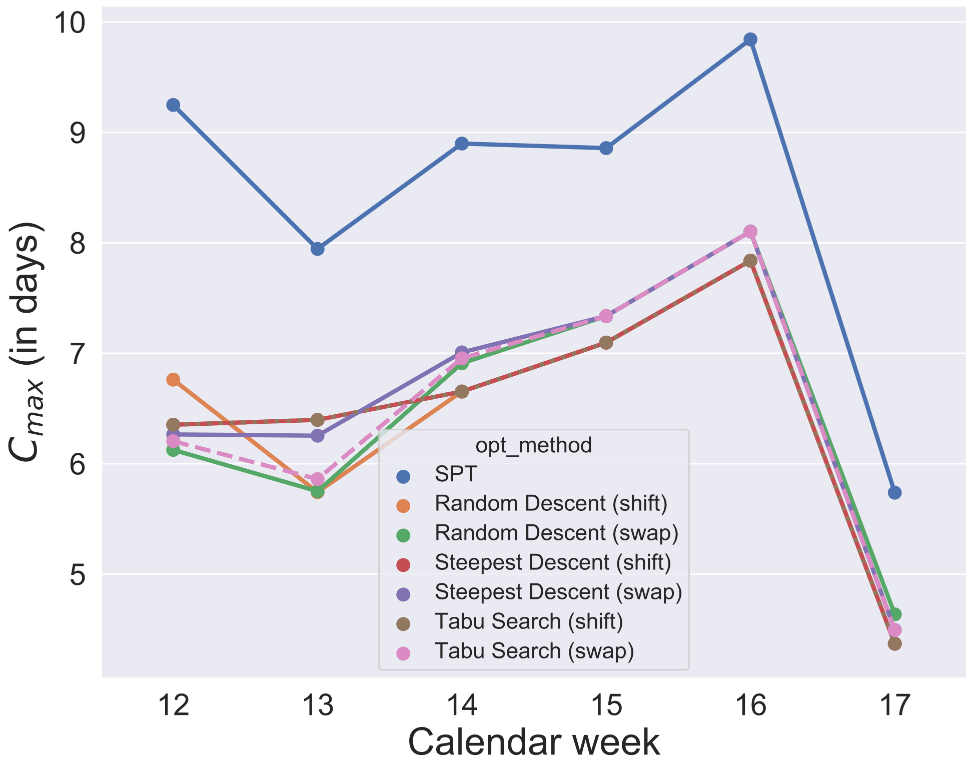

Figure 9. Optimization results for quantile 90%.

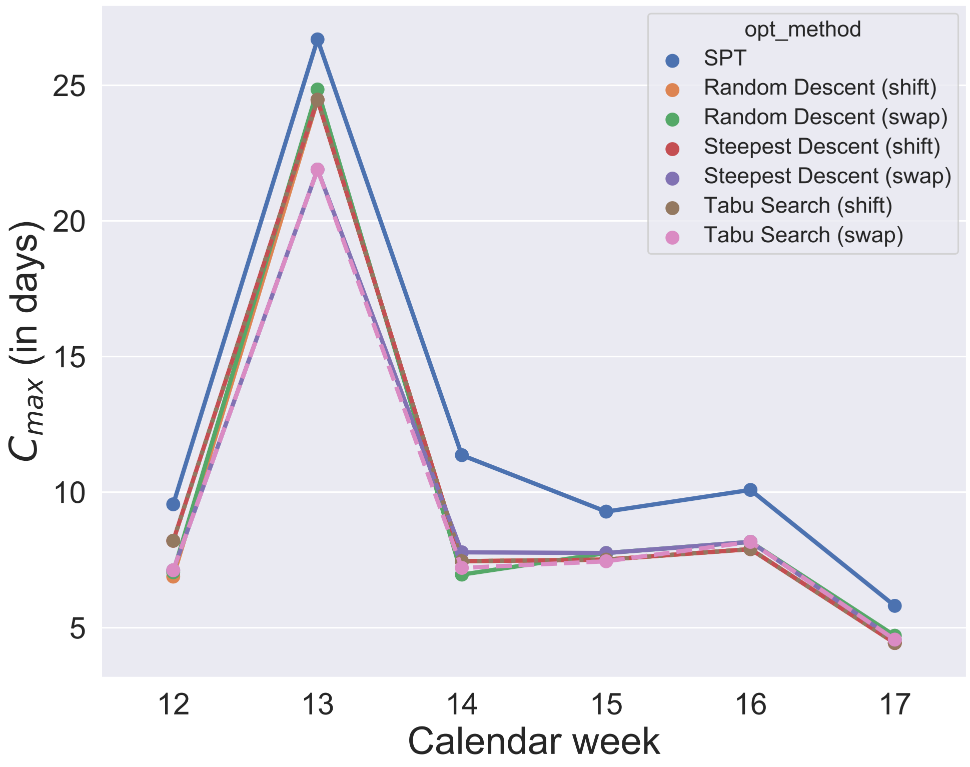

Figure 10. Optimization results for quantile 92%.

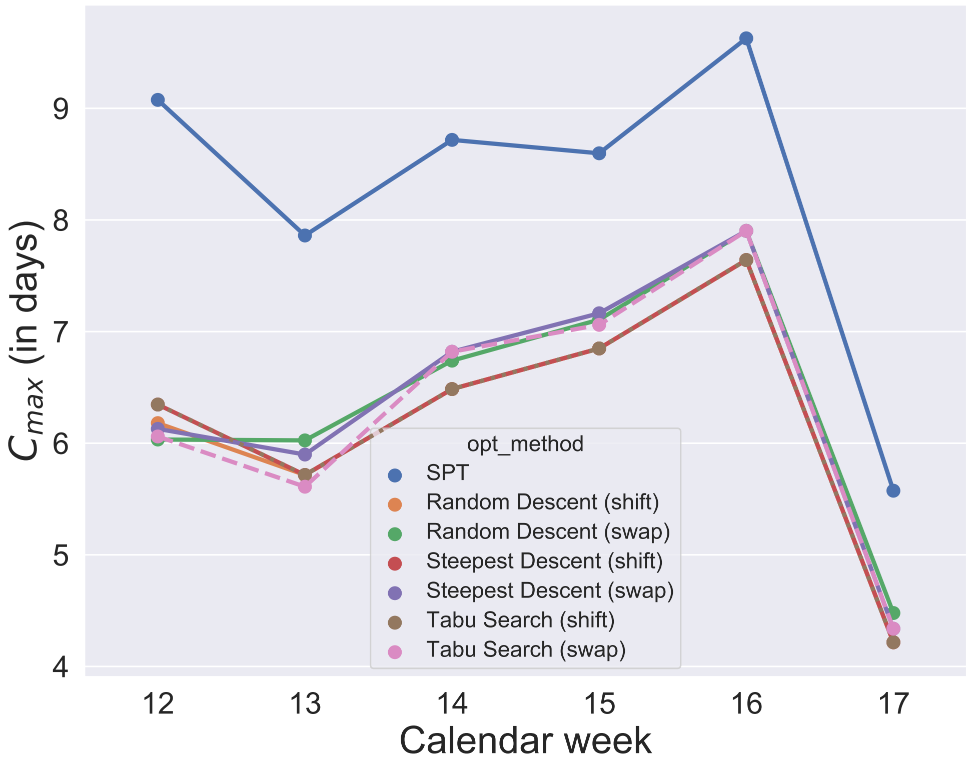

Figure 11. Optimization results for quantile 80%.

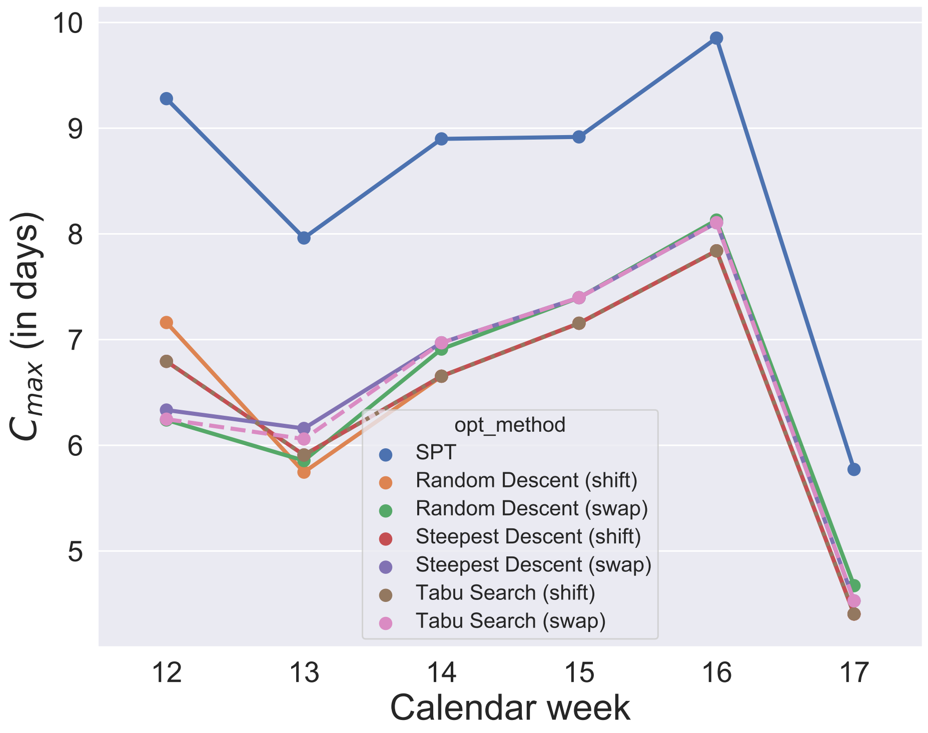

Figure 12. Optimization results for quantile 91%.

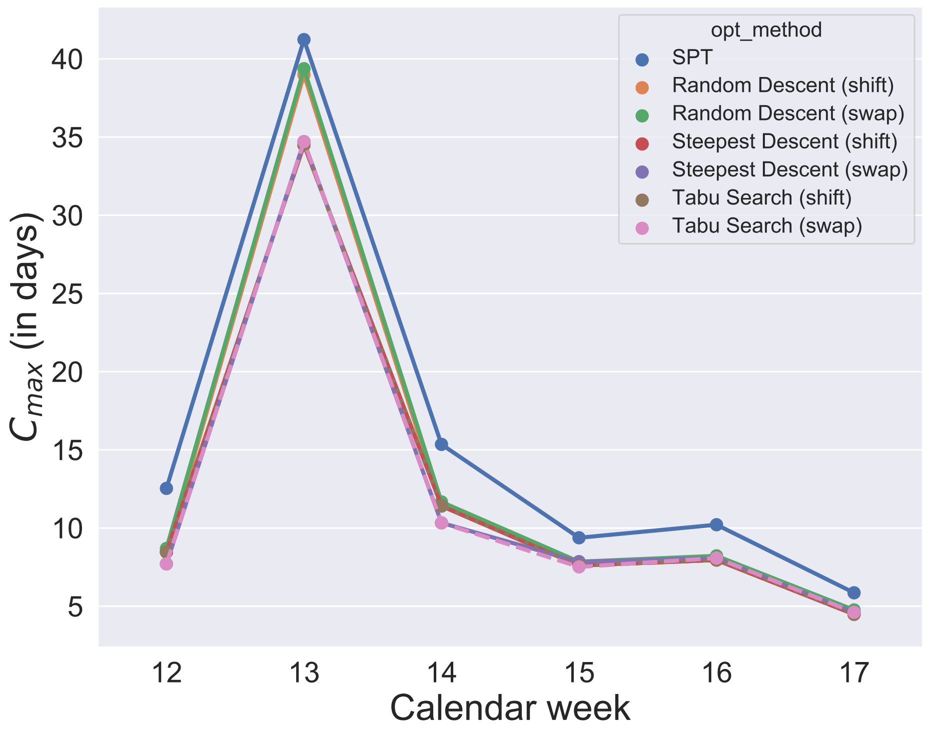

Figure 13. Optimization results for quantile 93%.

The largest computation time over all algorithms and weeks in optimization was about 4 min 45 s for a tabu search (swap) run and the minimum was less than 1 s for the calculation of SPT.

The results of C_{max} are visualized in Figures 8–13. Since the initial solution for the local search and tabu search algorithms is given by SPT, it is not surprising that, over all selected values for the quantile, the six metaheuristics provide better results than SPT. However, the difference between the C_{max} values and SPT is respectable. To show how the degree of usage for the machines increases in the production period by using metaheuristics rather than a static approach like SPT, Figure 14 compares one schedule of SPT and one schedule of a metaheuristic, namely the random descent (swap) exemplar for week 12. In contrast, the difference in makespan when comparing only the six metaheuristics for each week is minor. Hence, with this evaluation, we cannot determine which of the metaheuristics will perform best in the application case and which schedule should be applied in practice. Instead, we need simulation to see which solution of the metaheuristics is the most convenient and robust.

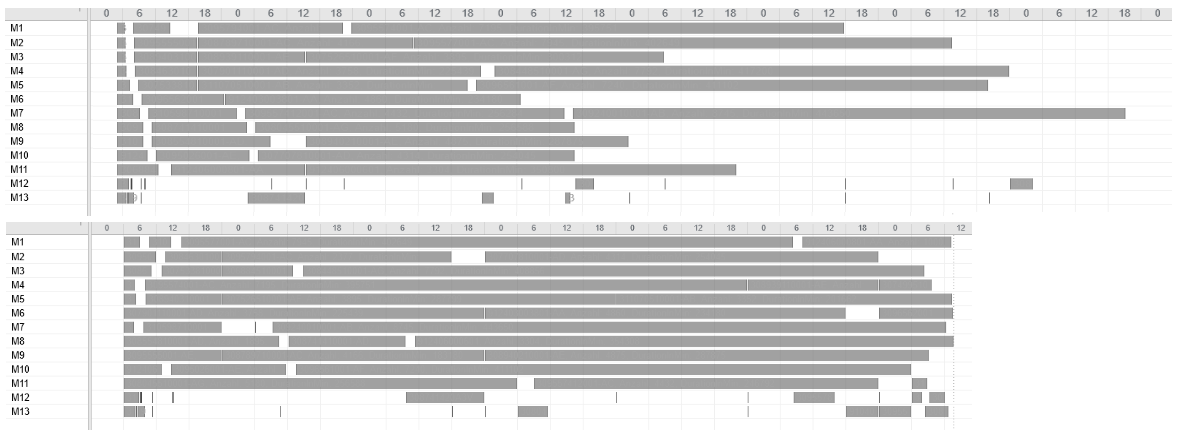

Figure 14. Comparison of schedules in MS Project for Shortest Processing Time (SPT) (**above**) and Random Descent swap (**below**) in calendar week 12 for quantile 91%.

Due to the high importance in our application case of fulfilling weekly demands as completely as possible, we also evaluate which quantile value for scrap performs best in optimization with regard to the makespan C_{max}. The results are also shown in Figures 8–13. It was found that the quantiles 60%, 80%, 90% and 91% provide similar good results for makespan. In contrast, comparing the C_{max} values of the 60%, 80%, 90% and 91% group of quantiles, and the 92% and 93% group shows large differences. The differences between 91% and 92% are also illustrated in an exemplary comparison of schedules in Figure 15. By increasing the value of the quantile by one percent in this exemplary week, the production volume $productionvolume_{ij}$ of one specific product increases from around 13 thousand parts to about 60 thousand parts and causes the extreme extension of the makespan shown in Figure 15. Considering the results of Figures 8–13, a production with a quantile higher than 91% seems uneconomical in this application case. To guarantee a sufficiently high degree of fulfilled orders and avoid extreme overproduction, we recommend the 91% quantile for the selected period of the application case.

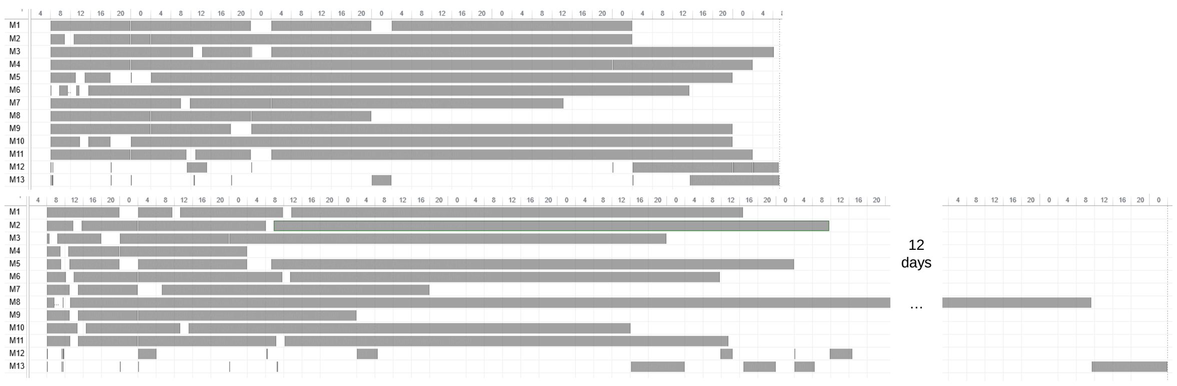

Figure 15. Comparison of schedules in MS Project for quantile 91% (**above**) and quantile 92% (**below**) in calendar week 13 with tabu search swap.

For comparing the fulfillment of orders and evaluating the makespan under near-realistic conditions for the different quantile values, we use simulation. Figures 16–19 show the variation in four different key indicators of the simulation results per quantile. For each algorithm and each calendar week, we run the simulation 40 times so that, per data point, we take the average of up to 280 simulation runs. Accordingly, for every figure, 280 key indicators are extracted from the simulation per data point. Only for Figure 17 are the SPT results excluded, mainly in order to highlight the possible achievable C_{max} values.

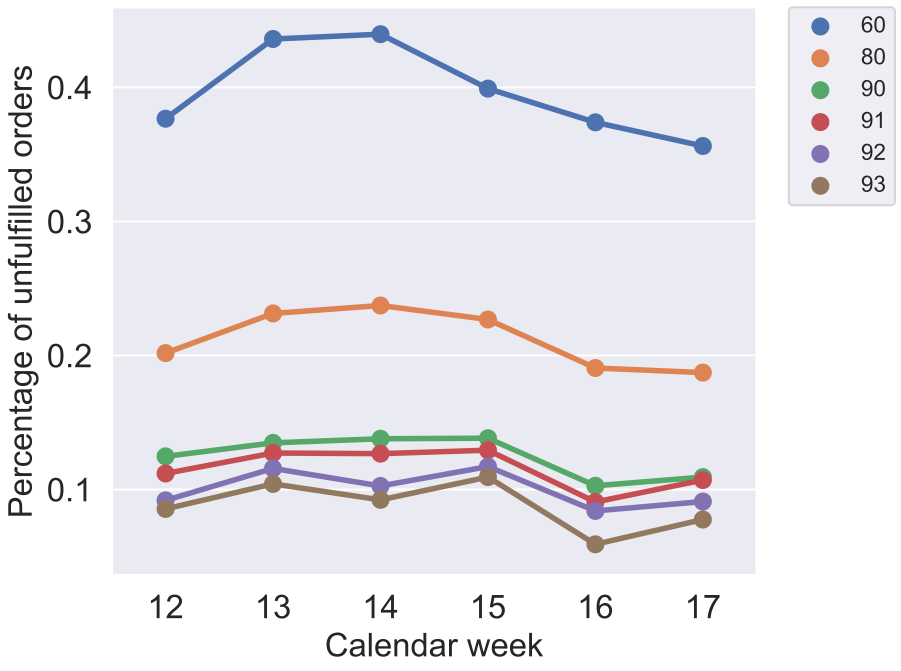

Figure 16. Average percentage of unfulfilled orders per quantile over all simulation runs.

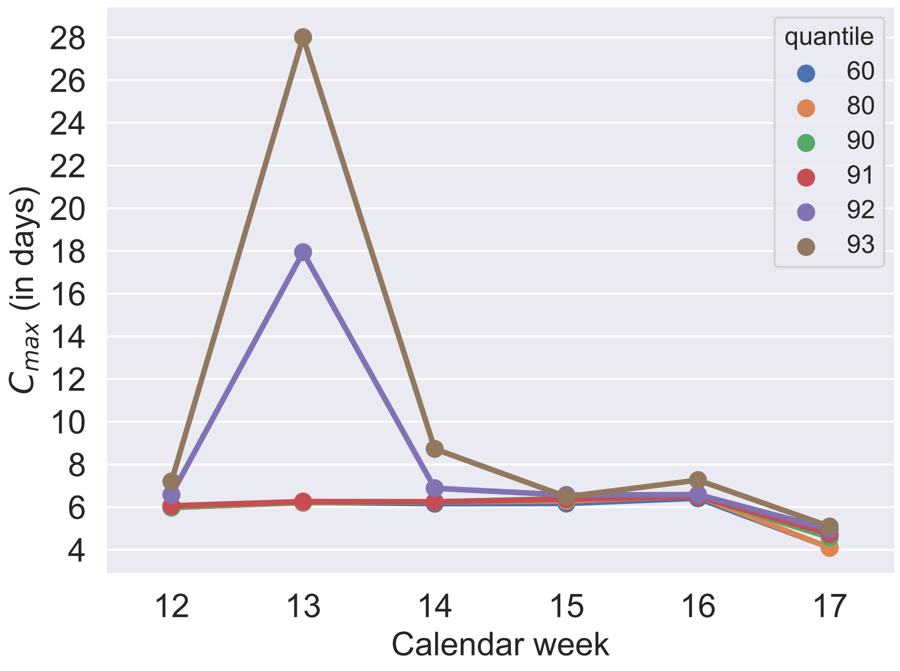

Figure 17. Average makespan of simulation runs over the six metaheuristics per quantile.

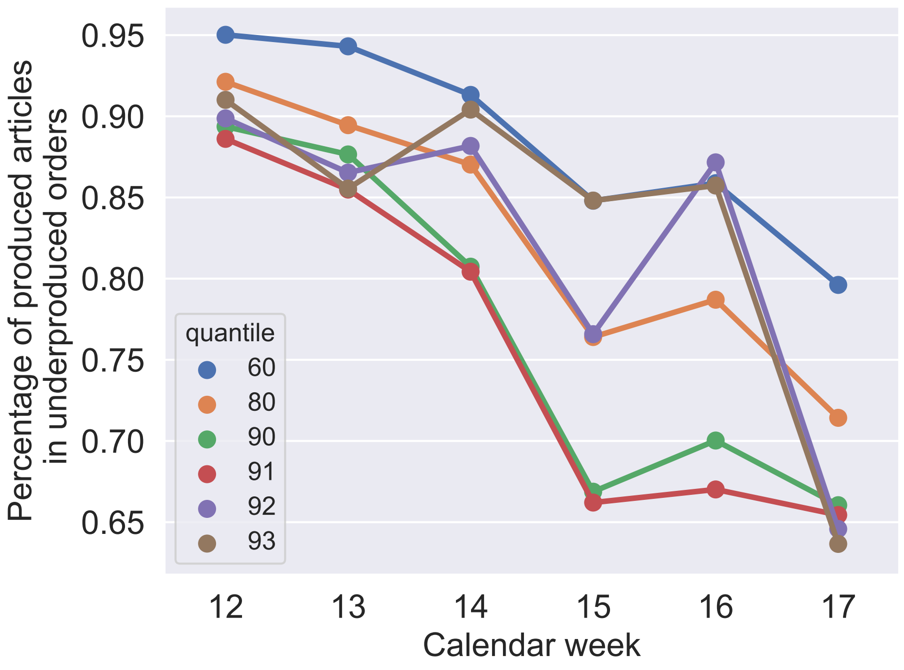

Figure 18. Average percentage of underproduction in orders producing too few items due to scrap.

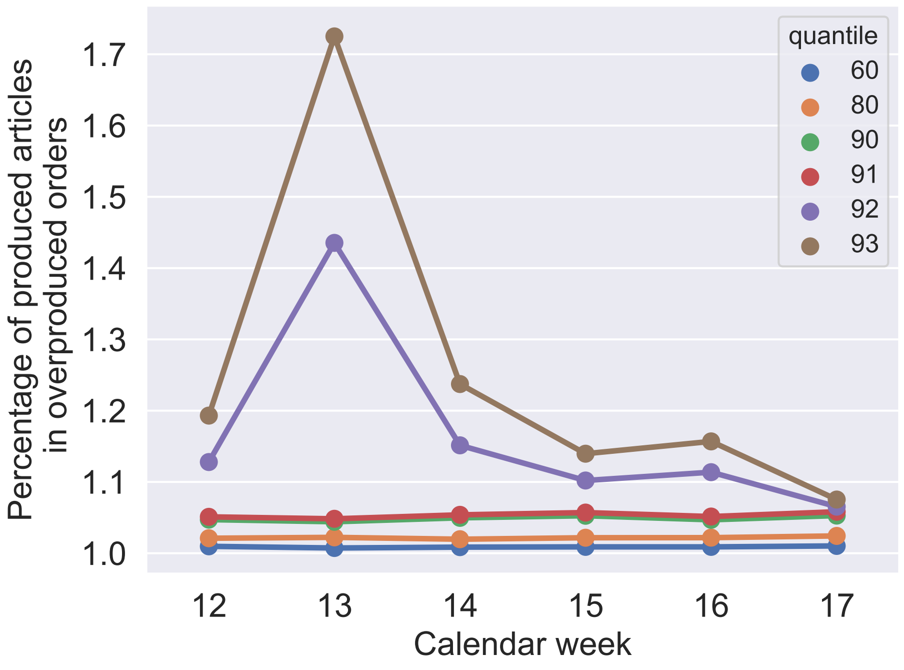

Figure 19. Average percentage of overproduction in orders producing too many items due to quantile method.

Figure 16 shows the ratio of orders that produce at least one less than the weekly demand. So, jobs are identified for that the weekly demand did not fulfil. This figure also validates our method to use scrap quantiles to avoid as many unfulfilled orders as possible. This demonstrates that the probability of unfulfilled orders decreases with increasing values for the different quantile values in the expected rates. The effects on the makespan show that the quantile value of 91% is preferable and can already be found in the optimization results of Figures 8–13 and are also validated by simulation with Figure 17.

Figures 16 and 19, when analyzed together, show that it is not realistic to reduce the probability of unfulfilled orders below 10 % without an extreme overproduction of orders. Furthermore, Figure 19 also confirms what could be expected from Figures 8–13, and Figure 17: the makespan increases, starting with the 92% quantile, because of significantly more overproduced parts in orders with overproduction.

In contrast, the results in Figure 18 do not indicate a clear correlation. For quantiles 60–91%, the rate of produced jobs decreases by increasing the quantile value, but quantiles 92% and 93% do not validate this assumption. Considering only this figure, quantiles 92% and 93% provide more preferable results. Although Figure 18 indicates that another quantile value should be chosen instead of the 91% quantile, the analyses of Figures 16, 17 and 19 confirm that the 91% quantile should be preferred for production planning in the application case. Because the most important indicators in our application case are the unfulfilled orders, in the following, we use the 91% quantile for our analysis.

To summarize the results under almost real conditions and to indicate which method returns the schedules with the best makespan and which schedule is most robust against uncertainties, e.g., in processing times or setup times, Figure 20 shows the makespans for week 12 of different schedules resulting from the seven algorithms analyzed by simulation. The box plots also indicate how the makespan varies in simulation runs of the same schedule. The borders of the boxes describe the first and third quartiles. Upper and lower limits show the minimum and maximum makespan of the simulation runs. Outliers are not included in Figure 20. For each schedule, 40 parallel replications are performed, requiring about 9 min. A good schedule should result in a small value for makespan and a small box, indicating robustness. The most robust solution in this week is provided by random descent (swap), but the average makespan is higher than the one provided by Steepest Descent (swap). Since the schedule resulting from Steepest Descent (swap) shows only small differences between the results of the different simulation runs and has a small average makespan for the production program of week 12, it should be selected for this week.

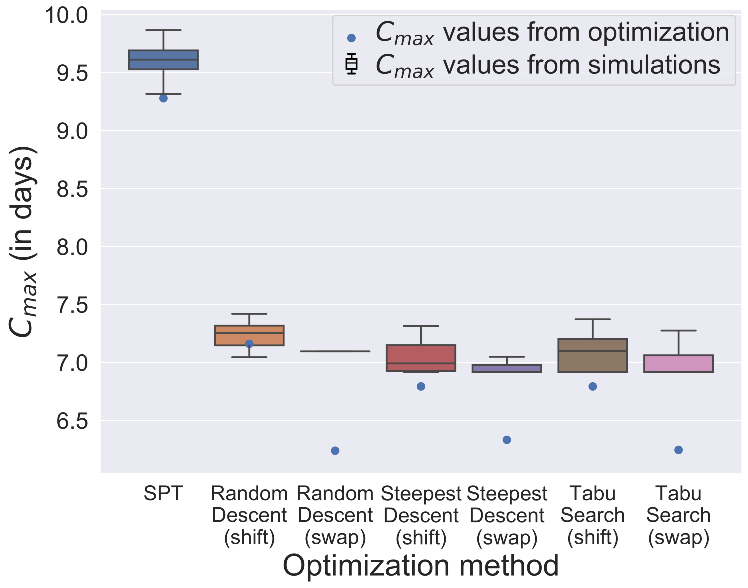

Figure 20. Comparison of simulation results per optimization method for calendar week 12 with quantile 91%.

9. Conclusions

In this paper, we solve a hybrid flow shop scheduling problem under uncertainties derived from a real industrial use case. It is shown that the combination of data analysis to estimate model parameters, heuristic and metaheuristic optimization and detailed stochastic simulation results in robust schedules that can be used in real systems. In addition, we have shown that, depending on the application, quantile analysis for scrap is an adequate method to reduce unfulfilled orders. Above a certain value of the quantile, their effect on preventing incompletely fulfilled demand positions is adversely affected by extreme overproduction and increased makespans. Thus, it is of high importance to analyze the quantiles and makespans depending on the application case.

Our approach can be developed further by using more efficient forecasting techniques to predict model parameters, by developing an upper bound for scrap rates, by additional heuristics and metaheuristics to generate schedules for the optimization model, and by combining the simulation model with additional techniques to optimize schedules. With the mentioned extensions, more efficient schedules are possible with regard to limiting overproduction, and achieving optimum production is possible. However, it is not expected that we will obtain significantly closer results to the simulation with modifications to the optimization methods because the simulation results are already very near to the results of the deterministic optimization.

Author Contributions: Conceptualization, C.S.; methodology, C.S.; software, C.S.; validation, C.S.; formal analysis, C.S.; investigation, C.S.; resources, P.B.; data curation, C.S.; writing—original draft preparation, C.S. and P.B.; writing—review and editing, C.S. and P.B.; visualization, C.S.; supervision, P.B.; project administration, C.S.; funding acquisition, P.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), grant number 276879186/GRK2193, within the Research Training Group GRK 2193 "Adaption Intelligence of Factories in a Dynamic and Complex Environment".

Acknowledgments: Special thanks goes to the company that generously provided us with the data for the use case and, in particular, to the production planer for their excellent cooperation. We want to express our great gratitude to our (former) student assistants– Kevin Fiedler, Nico Gorecki, Dominik Mäckel, Alina Esfahani and Emily Veuhoff—for their engagement and support with the project. In addition, we wish to thank the other members of the GRK 2193 for their productive collaboration.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

SPT	Shortest Processing Time
ECT	Earliest Completion Time

References

1. Borshchev, A.; Grigoryev, I. *The Big Book of Simulation Modeling*; AnyLogic: Chicago, IL, USA, 2014.
2. Ruiz, R.; Şerifoğlu, F.S.; Urlings, T. Modeling realistic hybrid flexible flowshop scheduling problems. *Comput. Oper. Res.* **2008**, *35*, 1151–1175. [CrossRef]
3. De Siqueira, E.C.; Souza, M.; de Souza, S.R. A Multi-objective Variable Neighborhood Search algorithm for solving the Hybrid Flow Shop Problem. *Electron. Notes Discret. Math.* **2018**, *66*, 87–94. [CrossRef]
4. Graham, R.L.; Lawler, E.L.; Lenstra, J.K.; Kan, A. Optimization and approximation in deterministic sequencing and scheduling: A survey. In *Annals of Discrete Mathematics, Proceedings of the Advanced Research Institute on Discrete Optimization and Systems Applications of the Systems Science Panel of NATO and of the Discrete Optimization Symposium Co-Sponsored by IBM Canada and SIAM Banff, Aha, Vancouver, BC, Canada, 1–31 August 1977*; Hammer, P.L., Johnson, E.L., Korte, B.H., Eds.; Elsevier: Amsterdam, The Netherlands, 1979; Volume 5, pp. 287–326. [CrossRef]

5. Pinedo, M. *Scheduling: Theory, Algorithms, and Systems*, 5th ed.; Springer: Cham, Switzerland; Heidelberg, Germany; New York, NY, USA; Dordrecht, The Nertherland; London, UK, 2016. [CrossRef]
6. Ruiz, R.; Vázquez-Rodríguez, J.A. The hybrid flow shop scheduling problem. *Eur. J. Oper. Res.* **2010**, *205*, 1–18. [CrossRef]
7. Ribas, I.; Leisten, R.; Framiñan, J.M. Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Comput. Oper. Res.* **2010**, *37*, 1439–1454. [CrossRef]
8. Komaki, G.M.; Sheikh, S.; Malakooti, B. Flow shop scheduling problems with assembly operations: A review and new trends. *Int. J. Prod. Res.* **2019**, *57*, 2926–2955. [CrossRef]
9. Jabbarizadeh, F.; Zandieh, M.; Talebi, D. Hybrid flexible flowshops with sequence-dependent setup times and machine availability constraints. *Comput. Ind. Eng.* **2009**, *57*, 949–957. [CrossRef]
10. Kurz, M.E.; Askin, R.G. Comparing scheduling rules for flexible flow lines. *Int. J. Prod. Econ.* **2003**, *85*, 371–388. [CrossRef]
11. Johnson, S.M. Optimal two- and three-stage production schedules with setup times included. *Nav. Res. Logist. Q.* **1954**, *1*, 61–68. [CrossRef]
12. Nawaz, M.; Enscore, E.E.; Ham, I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega* **1983**, *11*, 91–95. [CrossRef]
13. Low, C.; Hsu, C.J.; Su, C.T. A two-stage hybrid flowshop scheduling problem with a function constraint and unrelated alternative machines. *Comput. Oper. Res.* **2008**, *35*, 845–853. [CrossRef]
14. Naderi, B.; Ruiz, R.; Zandieh, M. Algorithms for a realistic variant of flowshop scheduling. *Comput. Oper. Res.* **2010**, *37*, 236–246. [CrossRef]
15. Burdett, R.L.; Kozan, E. Evolutionary algorithms for flow shop sequencing with non-unique jobs. *Int. Trans. Oper. Res.* **2000**, *7*, 401–418. [CrossRef]
16. Burdett, R.L.; Kozan, E. Evolutionary Algorithms For Resource Constrained Non-Serial Mixed Flow Shops. *Int. J. Comput. Intell. Appl.* **2003**, *3*, 411–435. [CrossRef]
17. Dios, M.; Fernandez-Viagas, V.; Framinan, J.M. Efficient heuristics for the hybrid flow shop scheduling problem with missing operations. *Comput. Ind. Eng.* **2018**, *115*, 88–99. [CrossRef]
18. Logendran, R.; deSzoeke, P.; Barnard, F. Sequence-dependent group scheduling problems in flexible flow shops. *Int. J. Prod. Econ.* **2006**, *102*, 66–86. [CrossRef]
19. Kaczmarczyk, W.; Sawik, T.; Schaller, A.; Tirpak, T.M. Optimal versus heuristic scheduling of surface mount technology lines. *Int. J. Prod. Res.* **2004**, *42*, 2083–2110. [CrossRef]
20. Geiger, M.J. Randomised Variable Neighbourhood Search for Multi Objective Optimisation. In Proceedings of the 4th EU/ME Workshop: Design and Evaluation of Advanced Hybrid Meta-Heuristics, Nottingham, UK, 4–5 November 2008; pp. 34–42.
21. Almeder, C.; Hartl, R.F. A Metaheuristic Optimization Approach for a Real-world Stochastic Flexible Flowshop Problem with Limited Buffer. *Int. J. Prod. Econ.* **2013**, *145*, 88–95. [CrossRef]
22. Burdett, R.L.; Kozan, E. Techniques to effectively buffer schedules in the face of uncertainties. *Comput. Ind. Eng.* **2015**, *87*, 16–29. [CrossRef]
23. Figueira, G.; Almada-Lobo, B. Hybrid simulation–optimization methods: A taxonomy and discussion. *Simul. Model. Pract. Theory* **2014**, *46*, 118–134. [CrossRef]
24. Juan, A.A.; Faulin, J.; Grasman, S.E.; Rabe, M.; Figueira, G. A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Oper. Res. Perspect.* **2015**, *2*, 62–72. [CrossRef]
25. Law, A.M. *Simulation Modeling and Analysis*, 5th ed.; Series in Industrial Engineering and Management; McGraw Hill: New York, NY, USA, 2015.
26. Murray, P.W.; Agard, B.; Barajas, M.A. Forecasting Supply Chain Demand by Clustering Customers. *IFAC-PapersOnLine* **2015**, *48*, 1834–1839. [CrossRef]
27. Lee, H.L.; Padmanabhan, V.; Whang, S. The bullwhip effect in supply chains. *Sloan Manag. Rev.* **1997**, *38*, 93–102. [CrossRef]
28. Kück, M.; Scholz-Reiter, B. A Genetic Algorithm to Optimize Lazy Learning Parameters for the Prediction of Customer Demands. In Proceedings of the 12th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 4–7 December 2013; Sayed-Mouchaweh, M., Wani, M.A., Eds.; IEEE Computer Society; IEEE: Piscataway, NJ, USA, 2013; pp. 160–165. [CrossRef]

29. Scholz-Reiter, B.; Kück, M.; Lappe, D. Prediction of Customer Demands for Production Planning—Automated Seelction and Configuration of Suitable Prediction Methods. *CIRP Ann.* **2014**, *63*, 417–420. [CrossRef]
30. Schumacher, C.; Buchholz, P.; Fiedler, K.; Gorecki, N. Local Search and Tabu Search Algorithms for Machine Scheduling of a Hybrid Flow Shop Under Uncertainty. In Proceedings of the 2020 Winter Simulation Conference, Orlando, FL, USA, 13–16 December 2020.
31. Van Hentenryck, P.; Bent, R. *Online Stochastic Combinatorial Optimization*; MIT Press: Cambridge, MA, USA, 2006.
32. Juan, A.A.; Panadero, J.; Reyes-Rubiano, L.S.; Faulin, J.; de la Torre, R.; Latorre, J.I. Simulation-Based Optimization in Transportation and Logistics: Comparing Sample Average Approximation with Simheuristics. In Proceedings of the 2019 Winter Simulation Conference, WSC 2019, National Harbor, MD, USA, 8–11 December 2019; pp. 1906–1917. [CrossRef]
33. Arthur, D.; Vassilvitskii, S. *k-means++: The Advantages of Careful Seeding*; Stanford University: Stanford, CA, USA, 2007.
34. Leskovec, J.; Rajaraman, A.; Ullman, J.D. *Mining of Massive Datasets*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2014. [CrossRef]