Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

UNIVERSITÄT HAMBURG
FAKULTÄT FÜR BETRIEBSWIRTSCHAFT
HBS HAMBURG BUSINESS SCHOOL
INSTITUT FÜR OPERATIONS RESEARCH

Selected topics on integrated production-scheduling and maintenance-planning problems

von

Sven Henrik Pries, M.Sc.,
geb. am 14.09.1990 in Hamburg

Arbeit vorgelegt als kumulative Disseration an der Fakultät für Betriebswirtschaft zur Erlangung des akademischen Grades
Doctor rerum oeconomicarum (Dr. rer. oec.)
gemäß Promotionsordung der Fakultät Betriebswirtschaft vom 09.07.2014

Lübeck, den 08.11.2022

Vorsitzende: Prof. Dr. Dorothea Alewell

Erstgutachter: Prof. Dr. Wolfgang Brüggemann

Zweitgutachter: Prof. Dr. Malte Fliedner

Datum der Disputation: 26.10.2022

Danksagung

Zu aller erst möchte ich Prof. Dr. Brüggemann für die Betreuung meiner Doktorarbeit danken. Ohne seine Anregungen wäre diese Arbeit nicht das, was sie am Ende geworden ist. Weiterhin möchte ich Prof. Dr. Fliedner für die Zweitbetreuung meiner Arbeit sowie Prof. Dr. Alewell als Vorsitzende der Prüfungskommission danken.

Mein weiterer Dank gilt meinen Kollegen am Institut für Operations Research. Im Speziellen möchte ich Celso Gustavo Stall Sikora und Andreas Geiger für all die Diskussionen und Anregungen über die letzten Jahre danken. Zudem möchte ich mich bei Christine Rodenbeck für die tatkräftige Unterstützung bei der Überarbeitung bedanken.

Zuletzt gilt ein ganz besonderer Dank meiner Familie und vor allem meiner Frau, die mich über die Zeit unterstützt hat und immer ein offenes Ohr für mich und meine Ideen hatte.

Contents

Chapter 1

Synopsis

1.1 Introduction

The scheduling of production jobs and the planning of maintenance activities on the used production capacities are among the most important planning tasks of modern industrial production. While the scheduling of jobs is necessary to produce goods, the insertion of maintenance activities into the schedule is necessary to maintain the high availability of the machines and hedge against delays due to random failures. Both, job processing and maintenance activities, are occupying machine time in the planning horizon. Because of the ongoing automation of production facilities and against the background of a continuous production, the conflict of both tasks becomes even more critical. The integrated planning of these tasks yields the best potential to cope with the interconnection of both decisions as pointed out in Pan et al. [2010]. However, Schmidt [2000], Ma et al. [2010] and Cui et al. [2018] discuss that the planning of maintenance activities or even the consideration of non-availability periods are less common in the scheduling literature. The latter can be seen as a way to integrate previously planned maintenance in the scheduling of production jobs. The stochastic case of the problem deals with random failures and the interrelation between the scheduling and maintenance decisions. Through this, the underlying problem becomes even more difficult to solve. Depending on the used modeling of the degradation and failure process, the problem is non-linear in nature and mainly meta-heuristic approaches are proposed to solve this problem in the literature. The preposition and discussion of mathematical programming approaches, especially mixed-integer programming (MIP), to this problem and the extension of related works are the focus of this thesis.
Five selected applications of the integrated production-scheduling and maintenance-planning problem (IPSMP) with random failures are discussed throughout this thesis. It is structured as follows. The related literature to this problem is outlined in section 1.2. Afterwards, a generalized form of the problem is shortly presented in section 1.3. Independent of machine environment or objective function, this section should show the main assumptions of the used stochastic process. Hereafter, the different approaches are shorty introduced. In section 1.4, a branch-and-price algorithm is proposed to minimize the expected makespan in a single-machine environment. The approach presented in section 1.5 solves the single-machine problem. The objective is to minimize the total weighted expected completion time. It is an extension to a branch-and-bound approach from the literature. Through a different view of the problem, further decision rules and problem-specific properties can be used to solve this problem. A decomposition approach for the minimization of the maximal expected completion time on identical parallel machines is discussed in section 1.6. A Benders' decomposition is used here as well as the proposed algorithm from section 1.4. This enables the utilization of classical Benders' cuts with a strong subproblem relaxation. Further, the difficult non-linear constraints of the problem can be shifted to the subproblems and become better manageable. In all previous works of this thesis, the uncertainty is encountered with the expected-value problem. Where the solution difference between this and a stochastic programming approach at the first two approaches is zero, it is not for the multi-machine problem. To consider this also, a stochastic programming approach to the cyclic permutation flowshop minimizing the maximal expected cycle time is discussed in section 1.7. A three-stage decomposition approach with tailored lower bounds and a Monte-Carlo simulation is used to solve the problem. In the works from section 1.6 and 1.7, a binary-mapping formulation is used to handle the non-linearities of the problem. A different formulation to enhance the solution process is proposed in section 1.8. The expected-value problem at a permutation flowshop environment minimizing the makespan for the expected number of

failures is solved here. In section 1.9, the synopsis is concluded. The full-length papers of all approaches can be found in chapter 2 to 6. An English and a German summary of this thesis are attached as Appendix 7.1 and 7.2, respectively. Hereinafter, a list of the articles (Appendix 7.3) and the applicant's contribution to them (Appendix 7.4) is outlined as well. The supplementary material of all works are online available through the link in Appendix 7.5.

1.2 Related literature

In the previous section, it is pointed out that considering non-availability periods is less common in scheduling literature. The scheduling with deterministic machine-availability constraints takes these periods into account. A survey of this deterministic case can be found in Schmidt [2000], [Ma et al., 2010] and Kaabi and Harrath [2014]. Various approaches can be named here for all kinds of production environments and objectives. The periods where the machine is non-available are often assumed to be fixed. Hence, the start and end times or the durations of the intervals are known and given beforehand. Therefore, no decisions on the intervals can be taken. This kind of models is suitable when maintenance decisions are taken beforehand in a sequential planning approach. Different kinds of problem-specific properties can be distinguished. To name a few, the number of intervals and their position, e.g., in multi-machine environments, can vary. Further, periodic cases are considered [Li et al., 2017; Krim et al., 2020]. Non-resumable [Adiri et al., 1989], semi-resumable [Beaton et al., 2016] and resumable cases [Lee, 1997] are discussed in the literature. In the first, the processing of jobs is not permitted to interfere with the non-availability interval. This is the most common variation especially for the advanced problem settings. In the resumable case, a job is allowed to be processed further after the period ends. In the semi-resumable case, the job can be continued but a penalty (e.g., in form of additional processing time or costs) occurs. Other approaches deal with no-wait assumptions in flowshop environments [Espinouse et al., 1999; Kubzin and Strusevich, 2005], setup times [Avalos-Rosales et al., 2018], learning effects [Vahedi-Nouri et al., 2013, 2014] and proportionally deteriorating jobs [Ji et al., 2006]. Another deterministic case of the problem is called the flexible case. Here, the number of intervals, the starting time or the duration are considered as decision variables. Therefore, it can be seen more as an integrated case than an interrelated case [Hadidi et al., 2012a]. Partial surveys for this case can be found in Hadidi et al. [2012a] and Sadiqi et al. [2018]. The consideration of this case is often achieved through the relaxation of the fixed assumptions discussed previously. Hence, maintenance periods need to be scheduled in a given time interval [Chen, 2008; Liao et al., 2017; Mosheiov et al., 2018] or periods need to be scheduled before the operational time since the last period exceeds a given threshold [Sun and Li, 2010]. Also deterioration of the maintenance activity duration [Mosheiov and Sidney, 2010; Wang et al., 2014] or the assumption that the machine(s) need(s) to be scheduled once in the planning horizon [Rebai et al., 2013] can be found. The intervals are considered here as a necessity. On the contrary, the scheduling of rate-modifying activities can be considered which reduces the impact of deteriorating jobs by resetting the deterioration process [Ji et al., 2013; Zhang et al., 2018]. The activities are not needed but can, for example, decrease processing times. This deterministic problem setting is closely related to the stochastic problem setting discussed throughout this thesis.

When random failures are considered, the stochastic case of the problem is observed. Maintenance activities can be scheduled here in the planning horizon to reduce the impact of these failures. Because of this decision on number, start and duration, the stochastic case

is also a flexible case. The integration of both decisions, production scheduling and maintenance planning, are considered here. A survey on related works can be found in [Hadidi et al., 2012a]. Away from the machine environments, objectives and additional properties, the approaches in the stochastic case can be distinguish best by the modeling of the failure process and the interconnection of the machine's condition with it. The problem setting of Cassady and Kutanoglu [2003] is mainly considered in the literature. Here, a minimal repair at failure is assumed. This means that the machine is set back to an operational state but without improving the machine's age. With this assumption the interval between two consecutive maintenance activities can be modeled as a non-homogeneous Poisson process. A Weibull hazard function is used here to ensure an increase in failure rate with increasing age. This assumption gives the model a non-linear character. The age can only be controlled with preventive maintenance activities. This problem setting forms the basis of this thesis and is further discussed in the generalized problem description in section 1.3. The single-machine environment is the most studied case. Here, full enumerative [Cassady and Kutanoglu, 2005], branch-and-bound algorithms [Wang and Liu, 2013] and meta-heuristic approaches [Sortrakul and Cassady, 2007] are proposed for the problem. The latter are further mostly used for bi- [Wang, 2013] as well as multi-objective problems [Yulan et al., 2008]. Besides the work of Hadidi et al. [2012b], nobody proposes mathematical programming in particular mixed-integer programming for this problem. No work is directed to the parallel-machine setting. For the flowshop environment, single-objective [Wang and Liu, 2016] and multi-objective [Cui et al., 2018] approaches can be found but less frequently than for the single-machine problem. These problems are also solved using meta-heuristic approaches.
Other approaches to model the integrated stochastic problem can also be found in the literature as well. These approaches differ in the underlying assumptions. Some assume a Markovian state degradation [Bajestani et al., 2014] or scenario-dependent health-state transition [Seif et al., 2019]. Also reliability-based approaches with exponential distribution [Mokhtari et al., 2012] or pessimistic value models with normal distribution [Shen and Zhu, 2019] can be found. Others assume deviating properties like imperfect repair [Zhang et al., 2021] or perfect repair [von Hoyningen-Huene and Kiesmüller, 2015] at failure.

1.3 Generalized problem description

A generalized description of the considered problem can be given as follows. A set of jobs J has to be scheduled on a set of machines I to minimize/maximize a given objective. The jobs are not allowed to interfere with each other. Therefore, they occupy machine time in the planning horizon. The machines are subjected to random failure during the processing of jobs. It is assumed that the probability of failures increases with the total operational time of a machine (further referred to as the age of the machine). Hence, the processing of jobs increases the machine age by the processing time of this job. Upon failure, an unplanned corrective maintenance (CM) is needed that delays the completion of a job by the duration of T^{cm}. A minimal repair is assumed at a CM, meaning that the machine is set back to an operational state without changing the age. With this assumption, the period of production can be expressed as a non-homogeneous Poisson process with a Weibull hazard function z with

$$z(t) = \frac{\beta}{\eta^\beta} t^{\beta-1} \tag{1.1}$$

with scale parameter $\eta > 0$ and shape parameter $\beta > 1$ [Cassady and Kutanoglu, 2003]. The latter ensures an increasing failure rate but also introduces non-linearities to the problem.

The expected number of failures ξ while a production period starting with a machine age (a_1) and ending with an age (a_2) can than be expressed as

$$\xi(a_2, a_1) = \left(\frac{a_2}{\eta}\right)^{\beta} - \left(\frac{a_1}{\eta}\right)^{\beta} = \int_{a_1}^{a_2} z(t)dt \tag{1.2}$$

as given in in Cassady and Kutanoglu [2005]. In combination with the Poisson distribution (poi), also the probability

$$Pr(\Omega, a_2, a_1) = poi(\Omega, \xi(a_2, a_1)) = \frac{\xi(a_2, a_1)^{\Omega} e^{-\xi(a_2,a_1)}}{\Omega!} \tag{1.3}$$

of a given number of failures Ω can be obtained.
Due to the assumption of minimal repair, the only way to reduce the age is to schedule preventive maintenance activities (PM) in the planning horizon. They set back the machine's age to an age of zero but occupy time in the planning horizon as well. The PM sets the machine to a state of as-good-as-new, also referred to as a perfect repair. The PM blocks the machine for production causing a delay of T^{pm} on its own. Because non-resumable jobs are assumed, these intervals are not allowed to interfere with the processing of jobs. Given all the data, the analytical optimal time τ^* between two consecutive PMs can be calculated as discussed in Cassady and Kutanoglu [2005] by maximizing the steady-state availability

$$A(\tau) = \frac{\tau}{\tau + \xi(\tau, 0)T^{cm} + T^{pm}} \tag{1.4}$$

which leads with differentiation to

$$\tau^* = \eta \left[\frac{T^{pm}}{T^{cm}(\beta - 1)}\right]^{\frac{1}{\beta}}. \tag{1.5}$$

This time cannot usually be met with the given assumptions and combinatorial approaches are needed to trade-off the delays of planned PMs and expected CMs. To do this and to incorporate it in alignment with the underlying scheduling problem, is the main focus of the problem studied herein.

1.4 A branch-and-price algorithm for the integrated production scheduling and maintenance planning on a single machine

In this article, a branch-and-price algorithm is proposed for a single-machine environment minimizing the expected makespan. The solution approach as well as the objective is novel to the IPSMP. A parameter value of $\beta = 2$ is assumed to obtain a well-manageable problem with a quadratic objective function. With the assumption that a PM is required at the beginning of the planning horizon, the single machine can be represented as a set of identical PM cycles. The assignment of jobs to these cycles results in delays either from the PM or CM part. Because of the special structure of this problem, it can be decomposed using a Dantzig-Wolfe decomposition. The master problem of the decomposition forms a set-partitioning formulation where each column represents a feasible assignment of jobs to a cycle. The objective is to find a set of columns that covers each job exactly once and minimizes the costs (meaning the expected delays). The set of feasible columns is huge compared to the number of constraints. Therefore, not all possible columns are used from the beginning and a column generation phase is applied. This generates promising columns (meaning columns with negative reduced costs) by solving a mixed-integer quadratic subproblem (MIQP). If no more columns can be found, the algorithm converges and has found a linear relaxation of the problem. This might incorporate non-integral decision variable values which is why a branching phase with ongoing column generation is needed to find the optimal integral solution. Different branching strategies are discussed in the paper as well as ways to accelerate the column generation phase. Two subproblem approaches are evaluated to generate multiple columns. First, a formulation as a shortest-path problem is proposed that is solved with dynamic programming. Second, an approach is discussed that utilizes feasible solutions found from a standard MIQP solver. All decomposition approaches outperform the monolithic model. While the shortest-path formulation shows good performance on small instances, it is outperformed by the MIQP approaches on larger ones. Both, the normal and the multi-column MIQP approach, show very good solving capabilities in favor of the multi-column approach. This approach wins most of the tested large instances. Additionally, all branch-and-price approaches show a good relaxation of the problem.

1.5 A branch-and-bound approach for integrated production scheduling and maintenance planning on a single machine

This article focuses more on an extension to an exact branch-and-bound approach rather than a mathematical programming approach for the problem. A single machine is considered and the objective is to minimize the total weighted expected completion time. In most literature for the IPSMP with random failures, and especially in the branch-and-bound approach of Wang and Liu [2013], a rank-based view of the problem is considered. A rank-based formulation, in contrast to a general precedence formulation, expresses the sequencing of jobs through an assignment to ranks ordered in a natural way (e.g., index-ascending order). The general precedence formulation ensures the sequencing through the pairwise predecessor-successor relationship of jobs. A MIP formulation is presented which can solve small-size instance. Two optimal decision rules are proposed in this work. The first, for the sequencing of jobs assigned to a PM cycle is an adaption of Wang and Liu [2013]. The second, for the sequencing of the PM cycles can be utilized with the general precedence formulation. It extends the decision rule for sequencing of chains proposed by Pinedo [2016]. With both decision rules, only the assignment of jobs to unordered cycles is of interest. This is solved using a branch-and-bound framework. The use of the decision rules decreases the tree size considerably. An analysis of the first decision rule under the assumption of the general precedence formulation reveals that parts of the pairwise comparison can be done in a preprocessing phase to accelerate the solution process. A lower bound as well as a heuristic rule to find fast upper bounds are discussed. The computational study is inspired by Wang and Liu [2013] and evaluates the performances of three versions of the proposed approach. First, the branch-and-bound algorithm without the heuristic rule is used. Second, a pure heuristic version of the approach is tested. At last, a combined 2-phase version is evaluated. It first uses the heuristic approach to find a good upper bound and than expands all pruned nodes to find the optimal solution. The results show that the heuristic version can find near optimal solutions in relatively short time. However, using this information in the exact approach does not pay-off because the calculation of the heuristic information increases the runtime significantly. The normal branch-and-bound approach shows the best performance and is able to solve up to 18 jobs to optimality and therefore shows slightly better results than Wang and Liu [2013].

1.6 Decomposition approach for integrated production and maintenance scheduling on parallel machines

This work extends partly the idea from section 1.4 to the parallel-machine environment with identical machines. The objective is to minimize the maximal expected completion time. The expected-value problem is considered and a shape parameter of $\beta = 2$ for all machines is assumed. In comparison to the single-machine approach, the quadratic terms are present in the constraints at this formulation and not in the objective function. The monolithic model however can be solved as a MIP using a binary mapping of ages to expected number of failures. By using a Benders' decomposition, the problem can be decomposed into a master and a subproblem. The master problem consists of the assignment of jobs to machines while the subproblem minimizes the expected makespan for the individual machines. Hence, the master problem is a MIP and the subproblem is a single-machine problem formulated as a MIQP. Combinatorial Benders' cuts are derived from the solution of the subproblem to exchange information. Further, a lower bound is given for the maximal expected completion time of the single-machine problem given an assignment. As the subproblem is the same problem as discussed in section 1.4, the subproblem can be decomposed using the approach proposed there. The single-column version of the proposed branch-and-price algorithm is used. This is because it showed comparable performance while not further expanding the callback depth of the algorithm. In summary, the proposed approach consists of an MIP Benders' master problem, a linear MIP Benders' subproblem (or also master problem of the branch-and-price) and a MIQP subproblem of the Dantzig-Wolfe approach. Because of the strong linear relaxation of the Benders' subproblem, classical Benders' cuts can be derived using the branch-and-price algorithm. These cuts are used as an extension to the combinatorial Benders' cuts. In a computational study, the performance of different configurations of the proposed ideas are evaluated. On small-size instances, the monolithic model is compared to the decomposition approaches. As for the latter, the pure Benders' decomposition, the Benders' with branch-and-price using just the combinatorial cuts and two version with classical Benders' cuts are used. The classical cuts are either added while adding the combinatorial cuts in a simultaneous fashion or, in another configuration, in a sequential approach. In the latter, the problem is solved only to its relaxation and all classical Benders' cuts are gathered. After adding them to the model formulation, the search is restarted, only adding combinatorial cuts. The results show that all decomposition approaches outperform the monolithic model. At a second test on large-size instances, it becomes evident that the sequential approach performs worse than the others with increasing number of machines. One reason could be that the root relaxation of this implementation has to be solved twice. The use of the branch-and-price approach accelerates the solution process on average. Compared to the normal Benders' decomposition where the performance deviates strongly, it performs better. Further, the simultaneously adding of classical cuts increases performance when solving instances with larger machine numbers and worsens it for small machine numbers.

1.7 Decomposition approach for integrated production scheduling and maintenance planning of a cyclic flowshop with random failures

Because most approaches from the literature to the multi-machine problem (also the approaches from section 1.6 and 1.8) consider the expected-value problem, a two-stage stochastic programming approach is considered in this work. The objective is to minimize the expected maximal cycle time of a cyclic permutation flowshop. Here, the starting and completion of the production plan is evaluated for each individual machine. The cyclic objective is novel to the problem and enables the planner to optimize small problem sizes and repeat them periodically to form larger schedules. In the stochastic programming approaches for the single-machine environment [Cassady and Kutanoglu, 2003], the failure scenarios are limited to only one failure per processed job. This limits the computational effort but might be less realistic. As stated in Cui et al. [2018], the multi-failure case can be considered by use of a Monte-Carlo simulation as done in this work. Cui et al. [2018], however, use a different approach for only the 0-1-failure case. For the multi-failure case, a decomposition heuristic with combinatorial cuts and problem-specific lower bounds is proposed in the present work. A Monte-Carlo approach is used to evaluate the value of the stochastic approach. The cuts are weak and a full enumeration of every feasible solution would be carried out if no upper and lower bound information are used. The first proposed lower bound approximates the actual value if the decisions on sequence and maintenance plans are given. The actual value means here the simulated value considering the stochastic programming approach. This bound uses the property that the objective value of the stochastic programming problem must be larger than the value of the expected-value problem. The first considers multiple scenarios whereby the latter only considers one scenario with expected data. Because the upper bound is just approximated here, this relation might not hold. Further, a lower bound is given for the actual value when the sequencing decision and no maintenance plan is given. To obtain this bound, the minimally expected processing times are scheduled. These consist of the deterministic processing times and the minimal expected delays. The latter considers an age of zero and no PMs in the schedule. With these two bounds, the problem is decomposed into a master problem generating sequencing decisions, a subproblem generating maintenance plans and a simulation model evaluating the actual value for both decisions. With given upper-bound information, feasible solutions can be pruned utilizing the lower bounds. The performances of these bounds are evaluated in a computational study. Here, the amount of feasible solutions is analyzed which can be cut away and does not need to be evaluated through the simulation. The results show that the first lower bound with both given decisions can be used more often. The second bound reduces more solutions if a sequence can be pruned because the maintenance plan evaluation can be excluded. However, this does not happen as frequently as seen with the first bound. Combining both lower bounds, instances with up to 7 jobs for the 2-machine environment and up to 5 jobs for the 3-machine setting can be solved. In addition to this, different ideas to accelerate the runtime of the simulation are discussed. Further, ways to reduce the possibility of wrongly pruned solutions are pointed out. These can be pruned due to the estimation quality of the simulation.

1.8 Mixed-integer formulations for the integrated production-scheduling and maintenance-planning problem in a flowshop

In the work in section 1.8, different mathematical programming formulations for the IPSMP with random failures on a permutation flowshop are discussed. The problem is based on the problem of Wang and Liu [2016] for which they propose a genetic algorithm. Contrary to the previous work, the objective of this problem is to minimize the makespan for the expected number of failures instead of the cycle time. Hence, instance-dependent starting ages are considered. The binary-mapping formulation to cope with the non-linearity is revised. Such a formulation is used in section 1.6 and 1.7 as well as in the MIP approach from section 1.5 to linearize the problem. The drawback of this formulation is the need for introducing every possible age and the corresponding expected number of failures to the model. Therefore, considerable computational effort is needed for the preprocessing as well as new binary variables for each combination. The binary variables are required for each possible age on every rank on every machine and this for both ages, the age prior to and that post to the job processing. In order to overcome this disadvantage, a constraint-based formulation is presented. For the case of shape parameter $\beta = 2$, this set of constraints is a simple linear inequality and the non-linearity in the formulation is eliminated. Unfortunately, BigM formulations are needed here for every job. For the general case of $\beta > 2$, the proposed set of constraints stays non-linear. However, it can be expressed through a lower approximation given all possible ages. Since the ages are discrete and known, the approximation is exact. The drawback of this approach is also the need to preprocess all possible ages and the introduction of constraints for each age-job-assignment combination. An iterative cut approach can be used here that iteratively solves the problem and adds the constraints for the ages which are present in the obtained solution. With each further considered age, the approximation of the expected number of failures becomes more precise. The algorithm converges if no new ages can be found. Note here that the binary mapping can be used for both shape parameter cases. To compare the performance of the constraint-based formulations to the binary mapping, a computational study on small-size instances is conducted. The results show that both constraint-based formulations can outperform the binary mapping on the 3-machine setting while reducing the preprocessing time. Further, it can be seen that these approaches perform worse on the 2-machine than on the 3-machine setting. An analysis shows that especially these approaches perform worse on instances where the spread between highest and lowest processing time is small. It seems that the binary mapping can cope better with this kind of instances. In a second test, the performance of the constraint-based approaches compared to a meta-heuristic approach is discussed. The test highlights the usability of this kind of approaches for the problem. At the special case of $\beta = 2$, the approach outperforms the considered genetic algorithm in terms of the best solution found. The same performance cannot be seen for the iterative approach where the genetic algorithm performs best. However, a comparable performance and further the information given by the MIP gap as a quality measurement of the solution are shown here.

1.9 Conclusion

In the review of related literature (section 1.2), it is outlined that different problem formulations regarding the integration of production scheduling and maintenance planning exist.

On the one hand, there is the deterministic case considering fixed or flexible intervals of non-availability. These intervals can be interpreted as PM activities which need to be scheduled. Closely related is the scheduling with job degradation, especially when combined with rate modifying activities. On the other hand, there is the stochastic case incorporating the flexible planning of maintenance activities to hedge against delays resulting from random failures or to ensure the availability of the observed system. With the assumption of minimal repair, the first can be modeled using a non-homogeneous Poisson process with a Weibull hazard function. The resulting model has a non-linear character and is mainly solved using meta-heuristic approaches. Less exact approaches are proposed in the literature. Also a lack of MIP approaches for the problem becomes visible. The works presented in this thesis propose approaches off the usually used methodologies to overcome these gaps.

All proposed approaches are concerned with the problem of integrated production scheduling and maintenance planning with random breakdowns given the underlying assumptions of section 1.3. The approaches differ mainly in scheduling environment, objective, solution approach, and consideration of the uncertainty. In section 1.4, a branch-and-price algorithm for the single-machine problem minimizing the expected makespan is proposed. The same environment but a different objective is discussed in 1.5. Here, a branch-and-bound algorithm and different versions of it are proposed to minimize the total weighted expected completion time. A decomposition approach to the identical parallel-machine environment minimizing the maximal expected completion time is discussed in 1.6. Here, a Benders' decomposition as well as the branch-and-price algorithm from section 1.4 are combined. The decomposition shifts the non-linearities to a well-manageable subproblem and enables the better use of classical Benders' cuts. The latter are derived from the good relaxation of the branch-and-price approach. The works in section 1.7 and 1.8 are both discussing a permutation flowshop setting. The first uses a decomposition heuristic to solve a stochastic programming problem minimizing the expected maximal cycle time. The second revises the linearization of the non-linear terms which are used throughout multiple approaches. An alternative formulation to the used binary mapping is given and also compared to the often used meta-heuristic approaches.

The presented approaches highlight the usability of exact mathematical programming to the problem and the expandability of proposed approaches to solve the integrated production-scheduling and maintenance-planning problem with random breakdowns.

References

Adiri, I., Bruno, J., Frostig, E. and Rinnooy Kan, A. H. G. [1989]. Single machine flow-time scheduling with a single breakdown, *Acta Informatica* **26**(7): 679–696.

Avalos-Rosales, O., Angel-Bello, F., Álvarez, A. and Cardona-Valdés, Y. [2018]. Including preventive maintenance activities in an unrelated parallel machine environment with dependent setup times, *Computers & Industrial Engineering* pp. 364–377.

Bajestani, M. A., Banjevic, D. and Beck, J. C. [2014]. Integrated maintenance planning and production scheduling with Markovian deteriorating machine conditions, *International Journal of Production Research* **52**(24): 7377–7400.

Beaton, C., Diallo, C. and Gunn, E. [2016]. Makespan minimization for parallel machine scheduling of semi-resumable and non-resumable jobs with multiple availability constraints, *INFOR: Information Systems and Operational Research* **54**(4): 305–316.

Cassady, C. R. and Kutanoglu, E. [2003]. Minimizing Job Tardiness Using Integrated Preventive Maintenance Planning and Production Scheduling, *IIE Transactions* **35**(6): 503–513.

Cassady, C. R. and Kutanoglu, E. [2005]. Integrating Preventive Maintenance Planning and Production Scheduling for a Single Machine, *IEEE Transactions on Reliability* **54**(2): 304–309.

Chen, J.-S. [2008]. Scheduling of nonresumable jobs and flexible maintenance activities on a single machine to minimize makespan, *European Journal of Operational Research* **190**(1): 90–102.

Cui, W., Lu, Z., Li, C. and Han, X. [2018]. A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops, *Computers & Industrial Engineering* **115**: 342–353.

Espinouse, M. L., Formanowicz, P. and Penz, B. [1999]. Minimizing the makespan in the two-machine no-wait flow-shop with limited machine availability, *Computers & Industrial Engineering* **37**(1-2): 497–500.

Hadidi, L. A., Turki, U. M. A. and Rahim, A. [2012a]. Integrated models in production planning and scheduling, maintenance and quality: a review, *International Journal of Industrial and Systems Engineering* **10**(1): 21.

Hadidi, L. A., Turki, U. M. A. and Rahim, M. A. [2012b]. Joint job scheduling and preventive maintenance on a single machine, *International Journal of Operational Research* **13**(2): 174.

Ji, M., He, Y. and Cheng, T. [2006]. Scheduling linear deteriorating jobs with an availability constraint on a single machine, *Theoretical Computer Science* **362**(1-3): 115–126.

Ji, M., Hsu, C.-J. and Yang, D.-L. [2013]. Single-machine scheduling with deteriorating jobs and aging effects under an optional maintenance activity consideration, *Journal of Combinatorial Optimization* **26**(3): 437–447.

Kaabi, J. and Harrath, Y. [2014]. A Survey of Parallel Machine Scheduling under Availability Constraints, *International Journal of Computer and Information Technology* pp. 238–245.

Krim, H., Benmansour, R., Duvivier, D., Aït-Kadi, D. and Hanafi, S. [2020]. Heuristics for the single machine weighted sum of completion times scheduling problem with periodic maintenance, *Computational Optimization and Applications* **75**(1): 291–320.

Kubzin, M. A. and Strusevich, V. A. [2005]. Two-machine flow shop no-wait scheduling with machine maintenance, *4OR* **3**(4): 303–313.

Lee, C.-Y. [1997]. Minimizing the makespan in the two-machine flowshop scheduling problem with an availability constraint, *Operations Research Letters* **20**(3): 129–139.

Li, G., Liu, M., Sethi, S. P. and Xu, D. [2017]. Parallel-machine scheduling with machine-dependent maintenance periodic recycles, *International Journal of Production Economics* **186**: 1–7.

Liao, W., Chen, M. and Yang, X. [2017]. Joint optimization of preventive maintenance and production scheduling for parallel machines system, *Journal of Intelligent & Fuzzy Systems* **32**(1): 913–923.

Ma, Y., Chu, C. and Zuo, C. [2010]. A survey of scheduling with deterministic machine availability constraints, *Computers & Industrial Engineering* **58**(2): 199–211.

Mokhtari, H., Mozdgir, A. and Kamal Abadi, I. N. [2012]. A reliability/availability approach to joint production and maintenance scheduling with multiple preventive maintenance services, *International Journal of Production Research* **50**(20): 5906–5925.

Mosheiov, G., Sarig, A., Strusevich, V. A. and Mosheiff, J. [2018]. Two-machine flow shop and open shop scheduling problems with a single maintenance window, *European Journal of Operational Research* **271**(2): 388–400.

Mosheiov, G. and Sidney, J. B. [2010]. Scheduling a deteriorating maintenance activity on a single machine, *Journal of the Operational Research Society* **61**(5): 882–887.

Pan, E., Liao, W. and Xi, L. [2010]. Single-machine-based production scheduling model integrated preventive maintenance planning, *The International Journal of Advanced Manufacturing Technology* **50**(1-4): 365–375.

Pinedo, M. L. [2016]. *Scheduling*, Springer International Publishing, Cham.

Rebai, M., Kacem, I. and Adjallah, K. H. [2013]. Scheduling jobs and maintenance activities on parallel machines, *Oper Res Int J* **13**(3): 363–383.

Sadiqi, A., El Abbassi, I., El Barkany, A. and El Biyaali, A. [2018]. Joint Scheduling of Jobs and Variable Maintenance Activities in the Flowshop Sequencing Problems: Review, Classification and Opportunities, *International Journal of Engineering Research in Africa* **39**: 170–190.

Schmidt, G. [2000]. Scheduling with limited machine availability, *European Journal of Operational Research* **121**(1): 1–15.

Seif, J., Dehghanimohammadabadi, M. and Yu, A. J. [2019]. Integrated preventive maintenance and flow shop scheduling under uncertainty, *Flexible Services and Manufacturing Journal* **153**(3): 534.

Shen, J. and Zhu, Y. [2019]. A parallel-machine scheduling problem with periodic maintenance under uncertainty, *Journal of Ambient Intelligence and Humanized Computing* **10**(8): 3171–3179.

Sortrakul, N. and Cassady, C. R. [2007]. Genetic algorithms for total weighted expected tardiness integrated preventive maintenance planning and production scheduling for a single machine, *Journal of Quality in Maintenance Engineering* **13**(1): 49–61.

Sun, K. and Li, H. [2010]. Scheduling problems with multiple maintenance activities and non-preemptive jobs on two identical parallel machines, *International Journal of Production Economics* **124**(1): 151–158.

Vahedi-Nouri, B., Fattahi, P., Rohaninejad, M. and Tavakkoli-Moghaddam, R. [2013]. Minimizing the total completion time on a single machine with the learning effect and multiple availability constraints, *Applied Mathematical Modelling* **37**(5): 3126–3137.

Vahedi-Nouri, B., Fattahi, P., Tavakkoli-Moghaddam, R. and Ramezanian, R. [2014]. A general flow shop scheduling problem with consideration of position-based learning effect and multiple availability constraints, *The International Journal of Advanced Manufacturing Technology* **73**(5-8): 601–611.

von Hoyningen-Huene, W. and Kiesmüller, G. P. [2015]. Evaluation of the expected makespan of a set of non-resumable jobs on parallel machines with stochastic failures, *European Journal of Operational Research* **240**(2): 439–446.

Wang, L.-Y., Huang, X., Ji, P. and Feng, E.-M. [2014]. Unrelated parallel-machine scheduling with deteriorating maintenance activities to minimize the total completion time, *Optimization Letters* **8**(1): 129–134.

Wang, S. [2013]. Bi-objective optimisation for integrated scheduling of single machine with setup times and preventive maintenance planning, *International Journal of Production Research* **51**(12): 3719–3733.

Wang, S. and Liu, M. [2013]. A branch and bound algorithm for single-machine production scheduling integrated with preventive maintenance planning, *International Journal of Production Research* **51**(3): 847–868.

Wang, S. and Liu, M. [2016]. Two-machine flow shop scheduling integrated with preventive maintenance planning, *International Journal of Systems Science* **47**(3): 672–690.

Yulan, J., Zuhua, J. and Wenrui, H. [2008]. Multi-objective integrated optimization research on preventive maintenance planning and production scheduling for a single machine, *The International Journal of Advanced Manufacturing Technology* **39**(9-10): 954–964.

Zhang, X., Wu, W.-H., Lin, W.-C. and Wu, C.-C. [2018]. Machine scheduling problems under deteriorating effects and deteriorating rate-modifying activities, *Journal of the Operational Research Society* **69**(3): 439–448.

Zhang, X., Xia, T., Pan, E. and Li, Y. [2021]. Integrated optimization on production scheduling and imperfect preventive maintenance considering multi-degradation and learning-forgetting effects, *Flexible Services and Manufacturing Journal* .

Chapter 2

Article 1: A branch-and-price algorithm for the integrated production scheduling and maintenance planning on a single machine

A branch-and-price algorithm for the integrated production scheduling and maintenance planning on a single machine

Sven Pries

Abstract

In this paper, a branch-and-price algorithm is presented to solve the integrated production-scheduling and maintenance-planning problem with random failures. The objective is to minimize the expected makespan in a single-machine environment. To reduce symmetry in the corresponding assignment problem of jobs to maintenance intervals, the problem is reformulated using a Dantzig-Wolfe decomposition, solving it with a branch-and-price approach. Both, objective and solution approach, are novel to this problem. To accelerate the solution procedure different subproblem approaches generating multiple columns are compared. The performance of these approaches and the overall branch-and-price approach are evaluated through a computational study.

Keywords: branch-and-price, single machine scheduling, preventive maintenance, random failures

1 Introduction

To ensure the high availability of production capacities in modern industrial production and to hedge against the impact of random failures, preventive maintenance (PM) activities need to be scheduled directly during working shifts if all day processing is assumed. PMs increase a machine's remaining useful life while job processing depletes it and increases the probability of failure. Both, PM and job processing, are competing for time in the planning horizon during which they occupy the machine [Cassady and Kutanoglu, 2003]. The problem studied in this work is the trade-off between the total PM time and the cumulative delays caused by random failures for a single-machine environment, minimizing the expected makespan.
Different ways to address periods of non-availability in the scheduling literature can be distinguished. A first example is the consideration of inserted PM activities as an interval, or often referred to as 'holes' as, for example, by Breit [2007], during which no job scheduling is allowed. If the deterministic case with fixed intervals is considered, meaning that times of non-availability and processing times are certain, the termination of these intervals comes from a previous sequential planning phase and is determined by a maintenance department before the production scheduling takes place. No interactions of both decisions are considered here. Comprehensive reviews by Lee et al. [1997], Sanlaville and Schmidt [1998], Schmidt [2000] and Ma et al. [2010] show that the problem where the intervals are fixed and known in advance has received a lot of attention. To point out the assignment character of this problem with makespan minimization, two works can be highlighted. Ángel-Bello et al. [2011a] study a single machine with sequence-dependent setup times and fixed periodic maintenance

activities. They assign jobs to the available intervals to minimize the makespan using a GRASP algorithm with a tabu search. Ángel-Bello et al. [2011b] solve the same problem with a mixed-integer programming model and discuss two ways, a valid inequality and the generation of a feasible starting solution, to reduce the running time of the model.
If the intervals are not fixed and the planner can decide on the number, the starting and end times or the duration of these intervals, the flexible deterministic case is analyzed. Here, partial surveys can be found in Hadidi et al. [2012a] and Sadiqi et al. [2018]. For the single-machine environment, Low et al. [2010] study both, the fixed and flexible problem. Zhao and Tang [2010] extend the problem with aging effects. Yang and Yang [2010] solve a single-machine problem that minimizes the sum of completion times and flexible maintenance activities with variable duration. Xu and Yin [2011] study the online and offline version of a single-machine problem. Their objective is to minimize the makespan while maintenance activities can be scheduled in a flexible manner. In case of the single-machine problem of Bock et al. [2012], the machine's condition decreases job-dependently. Full and partial maintenance cases are considered. The problem is solved using dynamic programming for various objectives. The problem of Kim and Ozturkoglu [2013] consists of a single machine with deteriorating processing times and is solved with a genetic algorithm. The deterioration process of this problem could be controlled by inserting PM activities. Xu et al. [2014] study a competitive two-agent single-machine problem with variable periodic PMs. For the problem of Mashkani and Moslehi [2016], the duration of a periodic PM depends on the previous working time. They solve it using a heuristic and a branch-and-bound approach. Wang et al. [2018] use a branch-and-price algorithm with an enhanced subproblem to solve the single-machine problem with linearly deteriorating jobs and flexible PM activities, minimizing the makespan.
Still, these approaches consider only a deterministic view on the maintenance. When delays resulting from random age-dependent failures are considered, the stochastic case of the problem is observed. To preserve the production from failures and also ensure the best production plan, the integrated scheduling of production jobs and flexible maintenance activities yields the potential to take into account these interdependencies. A survey to this problem is provided by Hadidi et al. [2012a]. For the single-machine environment, Cassady and Kutanoglu [2003] study the objective to minimize the total weighted expected tardiness. Only one failure while processing a single job is considered. They extend their work in Cassady and Kutanoglu [2005] for multiple failures per job, minimizing the total weighted expected completion time. Sortrakul and Cassady [2007] suggest a genetic algorithm for the weighted tardiness problem. Yulan et al. [2008] solve the single-machine problem with multiple objectives using again a genetic algorithm. Five objectives, including expected maintenance cost and expected makespan, are considered. Hadidi et al. [2011] study the single-machine problem with expected cost objective and extend their work to total weighted expected completion time in Hadidi et al. [2012b]. Wang [2013] suggests an evolutionary genetic algorithm to solve a bi-objective version of the problem using the total expected completion time and total expected failure time. A branch-and-bound algorithm for the problem with weighted expected completion time is presented by Wang and Liu [2013]. Abdelrahim and Vizvári [2017] solve the total expected completion time problem using a non-linear constrained 0-1-model that can be reduced to an unconstrained form. An approximation approach is discussed in Cui [2020] to deal with the uncertainty of the integrated problem in a robust way.
From the literature review, it becomes evident that the most common objective for the stochastic case of the integrated production-scheduling and maintenance-planning problem

for a single-machine environment is to minimize the total weighted expected completion time. Less work compared to the deterministic case is done for the expected makespan objective. Only the multi-objective approach of Yulan et al. [2008] considers this objective partially. To extend the work on the latter, is the focus of this paper. The resulting problem can be expressed as an assignment problem of jobs to the intervals between consecutive PM activities. To contribute to the solution approaches for this problem, a branch-and-price algorithm is suggested that decomposes the problem using Dantzig-Wolfe decomposition. Strategies to accelerate this solution procedure with enhanced subproblem approaches and different branching schemes are also discussed.
The paper is structured as follows. The problem is described in section 2 as an assignment problem with a quadratic objective function. The solution approach, presented in section 3, decomposes the problem into a master and a subproblem using different branching schemes to obtain an integral solution. This can be accelerated with various strategies. In section 4, the performance of the proposed approaches are evaluated by testing the normal and the improved decomposition approaches against the monolithic model on small-size instances. Further, the solving capabilities of the decomposition approaches are tested on medium- and large-size instances. The paper is summarized in section 5.

2 Problem description

The problem under consideration is a stochastic problem where a set of independent jobs J has to be scheduled on a single machine to minimize the schedule's expected makespan. This ranges from the beginning of production to the completion of the last job. Figure 1 displays a descriptive example of the problem.

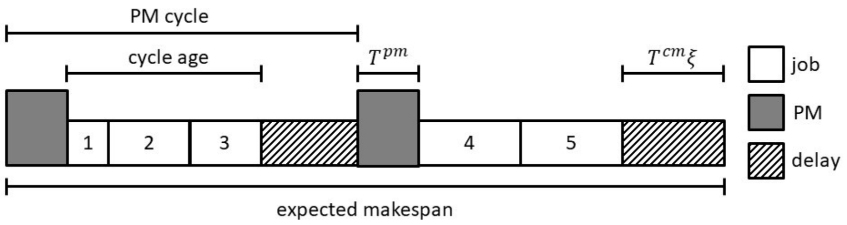

Figure 1: *Problem example*

Every job j has a processing time P_j and increases the machine age (or depletes the remaining total operational time) while processing. The machine can fail randomly throughout the production depending on its age, whereby an older machine is more likely to fail. If a failure occurs, the machine undergoes a corrective maintenance (CM) with a repair time of T^{cm}. In this case, the machine is minimally repaired, meaning the machine is set back to an operational state without changing the machine's age.
To reduce the delays caused by random failures, the machine can undergo PM activities (dark rectangles) which decrease the machine age but block the machine for production for a duration of T^{pm}. Balancing both delays is the target of this problem. The interval between two consecutive PMs is called a PM cycle i with I being the set of all such cycles. It is assumed that a PM cycle always starts with a PM activity and ends before the next. The total age of the machine at the end of a PM cycle i is the cycle age a_i. With a PM activity, the machine condition is set back to a state of as-good-as-new ($a_i = 0$). The accumulated delays caused by random failures during a PM cycle are represented by the shaded rectangles. The expectation of delays depends on a_i, i.e., the total job processing times assigned to the cycle.

The underlying stochastic process renews at the beginning of every cycle and is modeled as a non-homogeneous Poisson process with the hazard function

$$z(t) = \frac{\beta}{\eta^{\beta}} t^{\beta-1} \tag{1}$$

represented by a Weibull distribution with scale parameter $\eta > 0$ and the shape parameter $\beta > 1$ [Cassady and Kutanoglu, 2003]. The latter ensures the increase in occurrence of failures due to an increasing age. The expected number of failures is then represented by

$$\xi_i = \left(\frac{a_i}{\eta}\right)^{\beta} = \int_0^{a_i} z(t)dt \tag{2}$$

for every PM cycle i [Cassady and Kutanoglu, 2003]. In this work, it is assumed that $\beta = 2$ to yield a well-manageable quadratic formulation in the mathematical model, but the generalized problem formulation is provided in this and the following sections.
With this expectation, the optimal cycle age

$$a^* = \eta \left[\frac{T^{pm}}{T^{cm}(\beta - 1)}\right]^{\frac{1}{\beta}} \tag{3}$$

can be calculated analytically [Cassady and Kutanoglu, 2005]. This length, however, can only be met in special cases depending on the processing times and other solution approaches are needed.
Because the individual age and hence the expected duration of every PM cycle are independent from the job sequence as well as from the cycle sequence, just the set of jobs combined into a cycle is of interest. Hence, only the assignment of jobs to PM cycles must be considered. In this assignment problem, the set of jobs J should be distributed over the set of cycles I to minimize the expected makespan of the schedule. The binary decision variable

$$w_{ij} = \begin{cases} 1, & \text{if job } j \text{ is assigned to cycle } i \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

defines the assignment. To indicate whether jobs are assigned to a PM cycle or not, the binary cycle choice variable

$$cc_i = \begin{cases} 1, & \text{if there are jobs assigned to cycle } i \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

is used. The summarized notations for the assignment problem are shown in table 1.
With the given notation, the problem can be expressed as the monolithic model

$$\min \quad \sum_{i \in I} cc_i T^{pm} + a_i + \frac{T^{cm}}{\eta^{\beta}} a_i^{\beta} \tag{6}$$

$$\sum_{i \in I} w_{ij} = 1 \qquad \forall \quad j \in J \tag{7}$$

$$a_i = \sum_{j \in J} P_j w_{ij} \qquad \forall \quad i \in I \tag{8}$$

$$w_{ij} \leq cc_i \qquad \forall \quad i \in I, j \in J \tag{9}$$

Table 1: *Problem notation*

description	domain	type	meaning
I		index	set of cycles i
J		index	set of jobs j
P_j	(J)	data	processing time of job j
T^{pm}		data	duration of a PM
T^{cm}		data	duration of a CM
η		data	scale parameter of Weibull hazard function
β		data	shape parameter of Weibull hazard function
w_{ij}	(I, J)	binary	assignment of job j to PM cycle i
cc_i	(I)	binary	indicates if PM cycle i is used for scheduling
a_i	(I)	$\mathbb{R}^+$	age at the end of cycle i

$$cc_{i+1} \leq cc_i \qquad \forall \quad i \in I : i \neq |I| \tag{10}$$

$$w_{ij}, cc_i \in \{0, 1\} \qquad \forall \quad i \in I, j \in J \tag{11}$$

$$a_i \geq 0 \qquad \forall \quad i \in I. \tag{12}$$

The objective (6) is to minimize the expected makespan of the schedule. This is the sum of the expected duration of all PM cycles used. This is expressed as the combination of T^{pm}, the sum of included processing times and the accumulated delays through failure. Note here that the sum of all ages is the sum of all processing times and therefore a constant that can be omitted, but is further used for reasons of clarity. The delays due to failures result from the expected number of failures and the related failure time. Because $\beta = 2$ is assumed here, the expectation leads to a well-manageable quadratic objective function. However, also various shape-parameter values can be solved as a non-linear problem or, with the aid of a binary mapping from ages to expected numbers of failures, as a mixed-integer linear problem. Equation (7) ensures that every job is scheduled exactly once. The cycle's age is calculated by the sum of all processing times assigned to it with variable w_{ij}. If a PM cycle is used for the assignment, meaning that at least one w_{ij} is greater than 1, the cycle choice variable must also be 1 (9). To reduce symmetry in the model, inequality (10) orders the cycles and ensures that they are filled in an index-ascending order. (11) and (12) ensure the variable domains given in table 1.

3 Solution approach

3.1 Modeling

The assignment part of the problem can be reformulated with a Dantzig-Wolfe decomposition [Dantzig and Wolfe, 1960] and can be solved using column generation (CG) in a branch-and-price approach. For a primer in column generation see Desrosiers and Lübbecke [2005]. With this approach, the problem can be expressed as a set partitioning representation of the assignment part, the so-called master problem (MP). For the MP, each row of the technological matrix $A = (A_{jk})$ represent the coverage of a job in the set J. The set Ω contains the column indices k. Every column in (A_{jk}) represents a feasible combination of jobs included in the same PM cycle. Each is associated with a cost contribution c_k which is the expected makespan of the cycle. The rows ensure that each job has to be covered by exactly one pattern. The objective is to find a combination of columns, chosen with the binary variable x_k, that minimizes the expected makespan and fulfills the set-partitioning constraints.

$$\min \quad \sum_{k\in\Omega} c_k x_k \tag{13}$$

$$\sum_{k\in\Omega} A_{jk} x_k = 1 \qquad \forall \quad j \in J \tag{14}$$

$$x_k \in \{0,1\} \qquad \forall \quad k \in \Omega \tag{15}$$

The number of columns (every possible assignment of jobs to a particular cycle) in this formulation is huge in comparison to the number of rows (coverage constraint for each job). Solving such a problem can be difficult if all columns are considered. However, to form the optimal solution not all columns are needed. The method of column generation can be applied to the problem. It starts with a small set of columns at the MP and generates new promising ones during the process. Hence, just a subset $\overline{\Omega}$ of the existing column indices Ω is used. The MP with $\overline{\Omega}$ is called the restricted master problem (RMP) [Desrosiers and Lübbecke, 2005]. The initial $\overline{\Omega}$ includes only columns for two extreme cases. The first case is where all jobs are assigned to one cycle, which leads to an all-ones column. Second, the case with elementary columns where each job is assigned to its own cycle is included. To generate new columns, a subproblem (SP) is used which utilizes the constraints' dual values π_j of the current optimal linear programming (LP) solution of the RMP. The domain of x_k is therefore $\mathbb{R}^+$ in the RMP. The SP finds a pattern with negative reduced cost, if present, and the new pattern is added to $\overline{\Omega}$ of the RMP with its expected duration. The RMP is solved with the extended column set $\overline{\Omega}$ to obtain new dual values for the SP. The process continues until no patterns with negative reduced costs can be found anymore and the algorithm converges.
The SP chooses jobs which should be included in a new pattern to minimize the negative reduced costs. For the selection decision, the binary variable

$$y_j = \begin{cases} 1, & \text{if a job } j \text{ is included in the pattern} \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

is used. Expressions (17) to (20) represent the SP.

$$\min \quad T^{pm} + \frac{T^{cm}}{\eta^\beta} a^\beta - \sum_{j\in J} (\pi_j - P_j) y_j \tag{17}$$

$$c = T^{pm} + a + \frac{T^{cm}}{\eta^\beta} a^\beta \tag{18}$$

$$a = \sum_{j\in J} P_j y_j \tag{19}$$

$$y_j \in \{0,1\} \quad \forall \quad j \in J \tag{20}$$

The calculation of the negative reduced costs of a pattern (17) consists of three parts. First, every pattern has at least the duration of one PM. This and the additional delays resulting from the age-dependent expected number of failures worsen the negative reduced costs. With the choice of a job the costs are decreased by the difference of π_j and P_j of the corresponding job. Equation (18) calculates the expected cost of the new pattern that will be embedded in the RMP. This equation does not need to be in the solution process and can be calculated in a post-processing phase. The cycle age of the new pattern is computed in equation (19). The monolithic model can therefore be expressed as the combination of RMP and a sequence of SP combined within the Dantzig-Wolfe decomposition. At convergence of the CG, only an LP relaxation of the regular problem is obtained and x_k can take fractional values. To preserve the integrality condition of the regular problem, branching may be necessary. The algorithm branches on fractional solutions and continues the CG procedure for every new problem until an integral solution is reached [Barnhart et al., 1998]. With an incumbent integral solution and the lower bound given by the LP relaxation the search tree can be pruned. The algorithm terminates when every feasible integral solution is visited or excluded and therefore the optimal solution of the regular problem is obtained.

3.2 Branching

A natural way to perform branching is to branch on the number of used columns and therefore PM cycles in the planning horizon as given in Wang et al. [2018]. The total number of columns used can be represented by the sum of x_k. If this sum is fractional, there must be also fractional values of x_k included in the solution. The branching on the fractional sum via rounding down and up ensures that the sum of x_k should be either greater than or equal to $v_1 = \lceil \sum_{k \in \overline{\Omega}} x_k \rceil$ or less than and equal to $v_2 = \lfloor \sum_{k \in \overline{\Omega}} x_k \rfloor$ on the child nodes. After introducing

$$\sum_{k \in \overline{\Omega}} x_k \geq v_1 \quad \text{or} \quad \sum_{k \in \overline{\Omega}} x_k \leq v_2 \tag{21}$$

into the RMP and solve it, the corresponding dual variables ($\lambda_1 \geq 0$ and $\lambda_2 \leq 0$) need to be considered in the SP. Hence, the dual variables for the new constraints must be added to the objective function of the SP. This modifies (17) to

$$\min \quad T^{pm} + \frac{T^{cm}}{\eta^\beta} a^\beta - \lambda_1 - \lambda_2 - \sum_{j \in J} (\pi_j - P_j) y_j. \tag{22}$$

This changes the negative reduced cost and preserves the convergence of the SP. However, this branching strategy does not guarantee integrality because the integer bounds (21) can also be achieved with a combination of fractional x_k values as well.

To force the integrality of every x_k variable in this case, different branching approaches can be used. One way is to branch on the x_k variables in the RMP to ensure the fractional variable is either 1 or 0 as suggested in Wang et al. [2018]. Furthermore, it must be ensured that the subproblem does not generate columns which are fixed to zero in the RMP. However, according to Vanderbeck [2000], bounding a variable in this way leads to an unbalanced branching tree that has a negative impact on the performance. Better ways to branch are the schemes of Ryan and Foster [1981] or Vanderbeck [2011]. In this paper, the first scheme is used because of its specialized application for the set partitioning problem. It branches on job pairs of fractional columns in a conjunctive-disjunctive manner. The jobs of a pair are indexed by j_1 and j_2. On the left-hand branch the inequality

$$\sum_{k \in \overline{\Omega} \,:\, A_{j_1 k} = A_{j_2 k} = 1} x_k \geq 1 \tag{23}$$

for the RMP and equation

$$y_{j_1} = y_{j_2} \tag{24}$$

for the SP ensure that only columns are chosen after branching that include both jobs.
On the right-hand branch the opposite is enforced. The jobs of the pair cannot appear together in a single cycle. Equation (25) and (26) for the RMP and SP, respectively, exclude every column where both jobs are included and preserve this property in the generation of new columns.

$$\sum_{k \in \overline{\Omega} \;:\; A_{j_1 k} = A_{j_2 k} = 1} x_k \leq 0 \tag{25}$$

$$y_{j_1} + y_{j_2} \leq 1 \tag{26}$$

To find a suitable pair of jobs to branch on, a column with fractional x_k value nearest to 0.5 is chosen for evaluation in this paper. Here, a pair of jobs is chosen in index-ascending order that is not included in the previous branching process. With this branching strategy, the SP structure is changed due to the additional constraints.

3.3 Subproblem approaches

The mixed-integer quadratic problem (MIQP) of the SP has to be solved in every iteration which can be time consuming keeping in mind that finding just one solution for a MIQP can require considerable effort. The SP can also be solved with different approaches which enable the generation of multiple columns in the same iteration and add them to the RMP to accelerate the algorithm's CG step [Desaulniers et al., 2002]. Two approaches are discussed in this section. First, the SP is modeled as a shortest-path problem (SPP) and solved with dynamic programming. Second, a specialized MIQP approach which uses the solutions found by a solver during the solution process.
To transform the SP into a shortest-path problem, it can be described as a directed acyclic graph as discussed in Wang et al. [2018]. An illustrating example is given in figure 2. Not all possible edges are present due to the use of the second branching rule. Every numbered

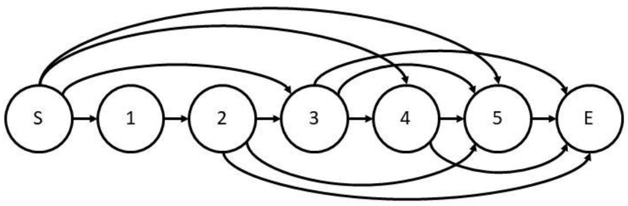

Figure 2: *Example of acyclic network*

node stands for a job chosen in the new pattern. Two dummy nodes (S and E) are added to the start and the end of the network. These nodes are not connected directly to ensure that at least one job is included in the cycle. Every edge is assigned the cost of choosing the job at the head of the edge. The initial cost at the start is T^{pm} minus the dual variable values λ_1 and λ_2 given from the first branching strategy. The incremental cost of every edge

$$c_{inc} = -(\pi_j - P_j) + \frac{T^{cm}}{\eta^\beta} a^\beta \tag{27}$$

consists of the negative fixed cost part $-(\pi_j - P_j)$ and the age-dependent positive cost part $\frac{T^{cm}}{\eta^\beta} a^\beta$.

This shortest-path problem can be solved by dynamic programming respecting all constraints introduced through the branching rules. Where the first branching rule leaves the structure of the problem unchanged, the second modifies this structure. Hence, the dynamic programming is more difficult to apply. At this, a state is defined by the current age and the successor set. The latter also defines the actions that can be taken in the current state.
To modify the network, the conjunctions and disjunctions of the second branching strategy can be modeled by use of edges. In the example in figure 2, node 1 and 2 must be assigned to the same cycle and therefore the only successor of node 1 is node 2. Nodes like this can be combined to reduce the number of nodes in the solution process. Further, the previous branching prescribes that node 2 and node 4 cannot be assigned to the same cycle. Hence, the direct edge from 2 to 4 is left out. However, this does not ensure the disjunction of both nodes because the path 2-3-4 is still possible. For these cases, the solution process must exclude these partial paths from further consideration.
To further accelerate the search procedure, ensuring to find the optimal solution, dominance properties can be introduced. These are valid for even larger shape parameters due to the monotonically increasing function of the age-dependent part. Note here that all conjunctive jobs are assumed to be already merged. If $-(\pi_j - P_j) \geq 0$ holds for a job j every path containing j is dominated by a similar path without this job because of the positive cost contribution. Hence, this job can be excluded from the node set in a pre-processing phase. Similar results can be seen with increasing age during the solution procedure. Given $-(\pi_j - P_j) < 0$ but $c_{inc} \geq 0$ the choice of this job at the current age increases the reduced cost and the age. A decision at a given state is dominated by another if reduced costs and age are greater after that decision. The path is dominated by similar paths without this job and every path with this partial path can be excluded. Paths with decisions which have higher reduced costs but a smaller age cannot be excluded from the candidate set and have to be further examined. A decision that has the lowest cost dominates the other decisions at a given state.
Every non-dominated path is saved for the generation of multiple columns. With this approach it is ensured that the optimal solution is found, but it is possible not to find the second best solution. The dominance rules can exclude these solutions from further consideration. For example, adding a job with $c_{inc} > 0$ to the optimal solution is of course not optimal, but the combined path can have a better objective function value than the second best solution found with the above approach. However, to find these solutions, more paths have to be evaluated. The proposed solution approach balances the solution time and quality making sure to find the optimal solution and therefore prove the convergence.
The second approach to generate multiple columns per iteration is to use the feasible, but not necessarily optimal solutions from the branch-and-cut procedure of the MIQP solver. At every feasible solution found, a callback is used to save this solution if the objective function value indicates a negative reduced cost. The procedure continues until the solver terminates at the proven optimal solution.
Both approaches generate multiple columns and add them to the RMP. With more columns the RMP becomes more difficult to solve. Therefore, only high quality columns are passed to the RMP when a given limit Ω_{lim} for generated columns is reached. In this case, only the Ω_{lim}-th best solutions are taken as new columns to the RMP. Otherwise, every found column is used.

4 Computational study

4.1 Data generation and design of experiments

To evaluate the performance of the branch-and-price algorithm and its different subproblem approaches three tests are performed.
First, the performance of these algorithms is compared to the benchmark approach 'MONO', which represents the monolithic model with the quadratic objective function. 'BP' is the branch-and-price approach using the normal quadratic subproblem, that is able to generate just one column. The approach with the shortest-path subproblem is called 'BP-SPP'. 'BP-MC' uses the MIQP solver to generate multiple columns out of feasible solutions found. This test is run on small-size instances, meaning the number of jobs $|J|$ per instance, with 10, 15 and 20 jobs. In a second test, only the performance of the branch-and-price approaches is evaluated on medium-size instances. They range from $|J| = 40$ to $|J| = 80$ with a step size of 10. At last, a test only evaluating the performance of 'BP' and 'BP-MC' is run on a test set with job numbers ranging from 100 to 250 with a step size of 25. For every job configuration, 30 random instances are generated which leads to 90 instances for the small-size test, 150 instances for the medium-size test and 210 instances for the large-size test. All instances and the corresponding results can be found in the supplementary material.
Furthermore, $T^{pm} = 5$, $T^{cm} = 15$, $\beta = 2$ and $\eta = 100$ are used as the same parameters as in Cassady and Kutanoglu [2005]. The processing times are randomly sampled from a discrete uniform distribution ranging from 1 to $\lfloor a^* \rfloor$. The rounding down of a^* reflects the assumption that the processing time of every individual job is less than the optimal interval and fits to the machine properties. Given (3) and the machine parameter, the processing times are sampled from the discrete uniform distribution $U[1, 57]$. The time limit for all approaches is set to 1800 seconds. The maximum of generated columns Ω_{lim} for both multi-column approaches is set to $|J|$. No warm-start approach is used in the branching phase.
To evaluate the performance, the total running time ('t-tot') is measured and it is counted how often the approach terminates in the given time limit ('term'). For the decomposition approaches, the relative time spent in the subproblem to the total running time ('rt-sp'), the number of nodes visited in the branching procedure ('bp-iter') and the total number of column generation iterations ('cg-iter') are recorded. Further, 'root' shows how often the decomposition approach solves the problem in the root node. To show the relaxation quality, a relative gap ('gap') is calculated from

$$\text{gap} = \frac{C^*_{max} - C^{relax}_{max}}{C^*_{max}} \tag{28}$$

using the objective function value at the root node C^{relax}_{max} and that of the optimal integral solution C^*_{max}. To compare the different approaches further, a competitive measurement is used. At 'win', a point is given to the approach that is the fastest for an instance, others are given zero points. In case that two approaches achieve identical fastest running times, both of them receive a point. The value reported in the 'win' column is then the number of points relative to the number of instances.

4.2 Results

The approaches are implemented in Python using Gurobi 9.1.0 as solver and the tests are run on an Intel(R) Core(TM) i5-6500 CPU machine at 3.2 GHz with 8 GB of RAM.

Table 2 to 4 show the results for all tests. The averages over all 30 instances are reported for the different measurements. For 't-tot', also the standard deviation is reported in brackets. Non-terminated runs were omitted from this calculation. All approaches work properly and find the proven optimal solution if they terminate in the given time limit.

Table 2: *Results for the competitive test at small-size instances*

$\|J\|$	sol	term	rt-sp	t-tot (σ)	win	bp-iter	cg-iter	gap	root
10	MONO	1	-	0.78 (0.55)	0	-	-	-	-
	BP	1	0.84	0.21 (0.11)	0	2.13	34.77	0.17e-03	0.47
	BP-SPP	1	0.52	0.05 (0.03)	1	2.07	10.43	0.17e-03	0.47
	BP-MC	1	0.85	0.18 (0.09)	0	2.07	28.00	0.17e-03	0.47
15	MONO	1	-	21.07 (40.73)	0	-	-	-	-
	BP	1	0.89	0.62 (0.32)	0	2.67	65.13	7.84e-05	0.33
	BP-SPP	1	0.68	0.34 (0.54)	0.9	2.87	18.33	7.84e-05	0.33
	BP-MC	1	0.89	0.49 (0.22)	0.1	2.47	49.70	7.84e-05	0.30
20	MONO	0.3	-	369.30 (532.12)	0	-	-	-	-
	BP	1	0.90	1.14 (0.59)	0.03	3.20	90.83	5.88e-05	0.23
	BP-SPP	1	0.73	1.04 (1.61)	0.67	3.33	20.63	5.88e-05	0.27
	BP-MC	1	0.90	0.91 (0.44)	0.30	3.27	68.27	5.88e-05	0.20

At the first test (table 2), 'BP', 'BP-SPP' and 'BP-MC' are able to solve every instance in the given time limit while 'MONO' is not. For the instances with 20 jobs, only 30% are solved to proven optimality. This may be due to the remaining high symmetry in this formulation which is broken by the decomposition approaches and also the weak relaxation of the formulation. Even at these instances the optimal solution is found by 'MONO', but not proven to be optimal in comparison to the decomposition approaches. The observations of 'term' are partially reflected by 'win'. Here, 'BP-SPP' wins most of the smaller instances. For the larger ones, a shift to 'BP' and particular 'BP-MC' can be seen. A stronger increase in subproblem running time compared to the other approaches might be a reason for this. Further, 'BP-SPP' performs better than 'BP-MC' in decreasing the number of CG steps which indicates that more columns are generated here. In comparison to the other approaches, it can be seen for 'BP-SPP' that the relative time spent in the subproblem is lower also at comparable total times. Hence, the RMP is harder to solve due to more columns included. Nearly 50% of the smaller instances and 20% of the larger ones can be solved in the root node of the branch-and-price algorithm. The low average count of branch-and-price iterations in both approaches also reflects this. The relative gap shows a tight relaxation due to the decomposition that is further reduced with instance size which is consistent with the literature [Desrosiers and Lübbecke, 2005].
In the first test, the superiority of the decomposition approaches over the monolithic becomes visible. For the second test, the performance of these approaches at larger instances is evaluated. The results are shown in table 3. The capabilities of the approaches to solve these instances decreases from 100% to approx. 90% for 'BP' and 'BP-MC' and 13% for 'BP-SPP' for instances with 80 jobs. The declined performance of 'BP-SPP' is also reflected by the increase in total running time and the decrease at 'win'. The 'BP-MC' wins more instances than 'BP', but the average running time is higher at 50 and 80 jobs. This means that 'BP-MC' is faster in most cases, but has extreme running times in the others. This becomes clear from the 'bp-iter' which is higher for the 'BP-MC'. Therefore, a poor path is

Table 3: *Results for the competitive test at medium-size instances*

$\lvert J \rvert$	sol	term	rt-sp	t-tot (σ)	win	bp-iter	cg-iter	gap	root
40	BP	1	0.93	7.23 (7.56)	0.1	9.87	227.47	1.20e-05	0.23
	BP-SPP	0.97	0.92	48.65 (78.59)	0	6.03	37.17	1.15e-05	0.21
	BP-MC	1	0.90	7.23 (11.16)	0.9	19.73	187.60	1.20e-05	0.27
50	BP	0.93	0.92	59.08 (158.57)	0.07	182.00	953.04	1.13e-05	0.25
	BP-SPP	0.87	0.93	299.47 (378.08)	0.03	126.92	226.00	1.11e-05	0.27
	BP-MC	0.93	0.88	82.86 (286.39)	0.90	333.43	916.86	1.13e-05	0.25
60	BP	0.80	0.93	13.20 (4.91)	0.04	4.67	344.00	5.61e-06	0.29
	BP-SPP	0.57	0.98	426.98 (365.94)	0	2.41	37.94	5.29e-06	0.41
	BP-MC	0.87	0.88	9.02 (4.07)	0.96	6.85	243.88	5.60e-06	0.23
70	BP	0.93	0.92	16.92 (5.57)	0.04	5.64	405.11	2.88e-06	0.32
	BP-SPP	0.43	0.99	668.17 (327.77)	0	2.23	37.00	3.10e-06	0.54
	BP-MC	0.93	0.88	9.77 (3.43)	0.96	5.21	276.50	2.88e-06	0.39
80	BP	0.87	0.89	25.02 (7.82)	0.07	9.77	540.77	3.46e-06	0.15
	BP-SPP	0.13	0.99	871.70 (528.01)	0	3.00	36.50	2.50e-06	0.50
	BP-MC	0.93	0.83	30.19 (56.14)	0.93	38.86	451.82	3.29e-06	0.14

chosen in the branching tree at these instances. Noticeable is the large average ‘bp-iter’ at $|J| = 50$ which are due to a small subset of the instances that can be solved within time limit but with a large branching tree. Similar instances exist for larger problem sizes but were excluded from the results due to time limit breach. The number of CG steps show similarities to the small instances. The multi-column approaches exhibit fewer iterations than ‘BP’ and the iteration count for ‘BP-SPP’ is smaller than for ‘BP-MC’ although Ω_{lim} is the same for both approaches. ‘BP-SPP’ generates more columns per iteration than ‘BP-MC’. However, comparing the relative subproblem times it becomes clear that ‘BP-SPP’ is taking more time for an iteration. The approach of the ‘BP-SPP’ implemented here scales poorly with increasing job number and is therefore omitted in the further test on large-size instances.

For the last test, only the performances of ‘BP’ and ‘BP-MC’ are compared (table 4). Both approaches show similar results as seen before. The termination rate decreases to approx. 70% for the largest instances. The total running time increases while the relative time of the SP decreases which means an increase in the RMP running time due to more added columns. The average running time of ‘BP-MC’ is less in every instance size. The same superior performance appears from ‘win’ where the approach wins 80% to 100% of the instances.

Overall, the results for the given parameter configuration show that the decomposition approaches outperform the monolithic approach and the improvements at the subproblem to use feasible found solutions of the MIQP solver are able to further accelerate the approach. The used ‘BP-SPP’ approach performs well at small problems but does not scale as well as the other approaches with an increasing number of jobs. Both, ‘BP’ and ‘BP-MC’, are capable to solve instances up to 250 jobs in the given time limit. The tight relaxation of the decomposition approaches can also be seen from the three tests.

Table 4: *Results for the competitive test at large-size instances*

$\|J\|$	sol	term	rt-sp	t-tot (σ)	win	bp-iter	cg-iter	gap	root
100	BP	0.70	0.91	33.48 (9.17)	0	5.67	609.48	1.10e-06	0.33
	BP-MC	0.73	0.85	19.64 (5.72)	1	6.27	450.14	1.31e-06	0.32
125	BP	0.77	0.86	55.78 (17.33)	0.13	13.26	886.65	1.57e-06	0.09
	BP-MC	0.77	0.77	34.87 (11.42)	0.87	12.30	644.04	1.48e-06	0.09
150	BP	0.67	0.82	74.85 (28.32)	0.14	18.70	1086.55	1.12e-06	0.10
	BP-MC	0.73	0.74	59.11 (58.16)	0.86	25.64	883.36	1.20e-06	0.09
175	BP	0.77	0.77	117.73 (42.61)	0.04	25.70	1398.39	1.23e-06	0.09
	BP-MC	0.77	0.68	73.73 (32.95)	0.96	19.09	1072.35	1.24e-06	0.09
200	BP	0.90	0.70	155.22 (55.25)	0.11	35.15	1674.48	7.33e-07	0.04
	BP-MC	0.87	0.62	105.38 (56.23)	0.89	26.54	1289.08	7.59e-07	0.08
225	BP	0.73	0.72	178.40 (78.67)	0.18	28.00	1875.50	3.77e-07	0.14
	BP-MC	0.67	0.65	122.00 (92.69)	0.82	21.10	1473.65	3.98e-07	0.10
250	BP	0.67	0.67	227.84 (90.00)	0.10	34.20	2174.05	4.27e-07	0
	BP-MC	0.67	0.59	150.26 (72.38)	0.90	24.50	1699.75	4.27e-07	0.20

5 Conclusion

In this paper, a branch-and-price algorithm is presented to solve the integrated production-scheduling and maintenance-planning problem with random failures for a single-machine environment, minimizing the expected makespan. Through the processing of jobs the machine age is depleted which increases the probability of random failures throughout the production. An occurring failure delays the completion of the current job whereas it does not affect the machine's age since a minimal repair at failure is assumed. The only way to reduce the age is by inserting PM activities to the schedule which set back the machine to a state of as-good-as-new. If a maintenance activity is scheduled no job can be processed. The assignment of jobs to the intervals between two consecutive PMs to trade-off the non-availability of the machine due to PMs and the impact of uncertain failure is the focus of this problem.
The problem can be solved using a monolithic mixed-integer formulation with a quadratic objective function. This formulation gives rise to decompose it based on the Dantzig-Wolfe decomposition into two models which can be solved using a branch-and-price approach. The resulting master and subproblem of the decomposed formulation are presented and both, different branching schemes as well as approaches to accelerate the CG phase, are presented. The performances of the proposed approaches are evaluated on three test sets. A set with small-size instances is used for the comparison of the monolithic model to the decomposition approaches. With a set of medium-size instances, the performance of the subproblem approaches are evaluated. The boundaries of the fastest approaches are tested on a set with large-size instances. The results show that the decomposition approaches outperform the monolithic model. Further, the subproblem acceleration of reusing found MIQP solutions ('BP-MC') outperforms all other decomposition approaches for larger instances.
A generalization of the found results at various parameter configurations would be of interest for further research, especially the performance of the different subproblem approaches

for larger shape-parameter values. The potential of other subproblem approaches or the improvement of the current can also be evaluated. Other accelerating strategies that, for example, improve the convergence of the column generation process or reduce the solution time of the master problem by removing unnecessary columns might be furthermore of interest for this problem.

References

Abdelrahim, E. H. and Vizvári, B. [2017]. Simultaneous Scheduling of Production and Preventive Maintenance on a Single Machine, *Arabian Journal for Science and Engineering* **42**(7): 2867–2883.

Ángel-Bello, F., Álvarez, A., Pacheco, J. and Martínez, I. [2011a]. A heuristic approach for a scheduling problem with periodic maintenance and sequence-dependent setup times, *Computers & Mathematics with Applications* **61**(4): 797–808.

Ángel-Bello, F., Álvarez, A., Pacheco, J. and Martínez, I. [2011b]. A single machine scheduling problem with availability constraints and sequence-dependent setup costs, *Applied Mathematical Modelling* **35**(4): 2041–2050.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P. and Vance, P. H. [1998]. Branch-and-Price: Column Generation for Solving Huge Integer Programs, *Operations Research* **46**(3): 316–329.

Bock, S., Briskorn, D. and Horbach, A. [2012]. Scheduling flexible maintenance activities subject to job-dependent machine deterioration, *Journal of Scheduling* **15**(5): 565–578.

Breit, J. [2007]. Improved approximation for non-preemptive single machine flow-time scheduling with an availability constraint, *European Journal of Operational Research* **183**(2): 516–524.

Cassady, C. R. and Kutanoglu, E. [2003]. Minimizing Job Tardiness Using Integrated Preventive Maintenance Planning and Production Scheduling, *IIE Transactions* **35**(6): 503–513.

Cassady, C. R. and Kutanoglu, E. [2005]. Integrating Preventive Maintenance Planning and Production Scheduling for a Single Machine, *IEEE Transactions on Reliability* **54**(2): 304–309.

Cui, W. [2020]. Approximate Approach to Deal with the Uncertainty in Integrated Production Scheduling and Maintenance Planning, *Journal of Shanghai Jiaotong University (Science)* **25**(1): 106–117.

Dantzig, G. B. and Wolfe, P. [1960]. Decomposition Principle for Linear Programs, *Operations Research* **8**(1): 101–111.

Desaulniers, G., Desrosiers, J. and Solomon, M. M. [2002]. Accelerating Strategies in Column Generation Methods for Vehicle Routing and Crew Scheduling Problems, *in* R. Sharda, S. Voß, C. C. Ribeiro and P. Hansen (eds), *Essays and Surveys in Metaheuristics*, Vol. 15 of *Operations Research/Computer Science Interfaces Series*, Springer US, Boston, MA, pp. 309–324.

Desrosiers, J. and Lübbecke, M. E. [2005]. A Primer in Column Generation, *in* G. Desaulniers, J. Desrosiers and M. M. Solomon (eds), *Column Generation*, Springer-Verlag, New York, pp. 1–32.

Hadidi, L. A., Turki, U. M. A. and Rahim, A. [2012a]. Integrated models in production planning and scheduling, maintenance and quality: a review, *International Journal of Industrial and Systems Engineering* **10**(1): 21.

Hadidi, L. A., Turki, U. M. A. and Rahim, M. A. [2011]. An integrated cost model for production scheduling and perfect maintenance, *International Journal of Mathematics in Operational Research* **3**(4): 395.

Hadidi, L. A., Turki, U. M. A. and Rahim, M. A. [2012b]. Joint job scheduling and preventive maintenance on a single machine, *International Journal of Operational Research* **13**(2): 174.

Kim, B. S. and Ozturkoglu, Y. [2013]. Scheduling a single machine with multiple preventive maintenance activities and position-based deteriorations using genetic algorithms, *The International Journal of Advanced Manufacturing Technology* **67**(5-8): 1127–1137.

Lee, C.-Y., Lei, L. and Pinedo, M. [1997]. Current trends in deterministic scheduling, *Annals of Operations Research* **70**: 1–41.

Low, C., Ji, M., Hsu, C.-J. and Su, C.-T. [2010]. Minimizing the makespan in a single machine scheduling problems with flexible and periodic maintenance, *Applied Mathematical Modelling* **34**(2): 334–342.

Ma, Y., Chu, C. and Zuo, C. [2010]. A survey of scheduling with deterministic machine availability constraints, *Computers & Industrial Engineering* **58**(2): 199–211.

Mashkani, O. and Moslehi, G. [2016]. Minimising the total completion time in a single machine scheduling problem under bimodal flexible periodic availability constraints, *International Journal of Computer Integrated Manufacturing* **29**(3): 323–341.

Ryan, D. and Foster, B. [1981]. An Integer Programming Approach to Scheduling, *Computer Scheduling of Public Transport* (1): 269–280.

Sadiqi, A., El Abbassi, I., El Barkany, A. and El Biyaali, A. [2018]. Joint Scheduling of Jobs and Variable Maintenance Activities in the Flowshop Sequencing Problems: Review, Classification and Opportunities, *International Journal of Engineering Research in Africa* **39**: 170–190.

Sanlaville, E. and Schmidt, G. [1998]. Machine scheduling with availability constraints, *Acta Informatica* **35**(9): 795–811.

Schmidt, G. [2000]. Scheduling with limited machine availability, *European Journal of Operational Research* **121**(1): 1–15.

Sortrakul, N. and Cassady, C. R. [2007]. Genetic algorithms for total weighted expected tardiness integrated preventive maintenance planning and production scheduling for a single machine, *Journal of Quality in Maintenance Engineering* **13**(1): 49–61.

Vanderbeck, F. [2000]. On Dantzig-Wolfe Decomposition in Integer Programming and ways to Perform Branching in a Branch-and-Price Algorithm, *Operations Research* **48**(1): 111–128.

Vanderbeck, F. [2011]. Branching in branch-and-price: a generic scheme, *Mathematical Programming* **130**(2): 249–294.

Wang, S. [2013]. Bi-objective optimisation for integrated scheduling of single machine with setup times and preventive maintenance planning, *International Journal of Production Research* **51**(12): 3719–3733.

Wang, S. and Liu, M. [2013]. A branch and bound algorithm for single-machine production scheduling integrated with preventive maintenance planning, *International Journal of Production Research* **51**(3): 847–868.

Wang, T., Baldacci, R., Lim, A. and Hu, Q. [2018]. A branch-and-price algorithm for scheduling of deteriorating jobs and flexible periodic maintenance on a single machine, *European Journal of Operational Research* **271**(3): 826–838.

Xu, D., Liu, A. and Yang, D.-L. [2014]. Mathematical Programming Models for Competitive Two-Agent Single-Machine Scheduling with Flexible Periodic Maintenance Activities, *Arabian Journal for Science and Engineering* **39**(5): 3715–3722.

Xu, D. and Yin, Y. [2011]. On single-machine scheduling with flexible maintenance activities, *The International Journal of Advanced Manufacturing Technology* **56**(9-12): 1139–1145.

Yang, S.-J. and Yang, D.-L. [2010]. Minimizing the total completion time in single-machine scheduling with aging/deteriorating effects and deteriorating maintenance activities, *Computers & Mathematics with Applications* **60**(7): 2161–2169.

Yulan, J., Zuhua, J. and Wenrui, H. [2008]. Multi-objective integrated optimization research on preventive maintenance planning and production scheduling for a single machine, *The International Journal of Advanced Manufacturing Technology* **39**(9-10): 954–964.

Zhao, C.-L. and Tang, H.-y. [2010]. Single machine scheduling with general job-dependent aging effect and maintenance activities to minimize makespan, *Applied Mathematical Modelling* **34**(3): 837–841.

Chapter 3

Article 2: A branch-and-bound approach for integrated production scheduling and maintenance planning on a single machine

A branch-and-bound approach for integrated production scheduling and maintenance planning on a single machine

Sven Pries

Abstract

In this paper, the integrated production-scheduling and maintenance-planning problem for a single-machine environment with random failures is considered. The objective is to minimize the total weighted expected completion time. The processing of jobs depletes the machine's condition which is then more likely to fail randomly. Here, corrective maintenance is needed. Preventive maintenance can be scheduled to reduce the probability of failures. To solve the problem, a branch-and-bound approach is proposed in this paper that incorporates expanded decision rules and problem-specific properties. This can be utilized due to a general precedence formulation which is uncommon to this problem. The performances of three variants of the proposed approach are evaluated in a computational study and show good solving capabilities.

Keywords: scheduling, maintenance, branch-and-bound, single machine, random failures

1 Introduction

In the production scheduling literature, the machines, on which jobs are to be scheduled, are often assumed to be continuously available [Schmidt, 2000]. For a real industrial case, this may not be true [Ma et al., 2010]. The availability of machines can be reduced due to lunch breaks, scheduled preventive maintenance or random failures. Schedules which take this kind of non-availability into account are needed to find better solutions for more realistic cases. Much work has been done for the deterministic case of this planning problem which is the so-called production scheduling with availability constraints. The single-machine environment can be seen as a building block for more complex machine environments [Schmidt, 2000] and also receives most attention in the literature.
Within this class of problems, the non-availability periods can be a fixed time interval. These for example, are known in advance through a previous planning phase that comes from a sequential planning approach. Comprehensive reviews by Lee et al. [1997], Sanlaville and Schmidt [1998], Schmidt [2000] and Ma et al. [2010] show that this case is well studied. Interesting branch-and-bound approaches for this case can be found in the papers of Qi et al. [1999] and Batun and Azizoğlu [2009]. Another possibility is to integrate the decisions linked to the periods into the planning process benefiting from a better interaction of both decisions. The time interval is now a decision variable that can differ in position, size and number depending on the problem. Less work is done for this flexible case. Partial surveys are provided by Hadidi et al. [2012a] and Sadiqi et al. [2018]. For the single-machine problem, Low et al. [2010] study fixed and flexible cases. The problem of Zhao and Tang [2010]

includes aging effects and flexible maintenance. Yang and Yang [2010] examine a single machine with variable duration of availability constraints to minimize the total completion time. Xu and Yin [2011] solve the online and offline version of a single-machine problem with flexible maintenance activities minimizing the makespan. In Bock et al. [2012], a single-machine scheduling with machine deterioration with full and partial maintenance is solved using dynamic programming. Kim and Ozturkoglu [2013] study a problem with deteriorating processing times. Here, multiple preventive maintenance intervals can be scheduled to reduce the deterioration. They solve the problem with a genetic algorithm. Xu et al. [2014] assess a competitive two-agent single-machine scheduling with flexible periodic maintenance activities. A single-machine problem where the duration of the flexible periodic non-availability intervals correspond to the working time is studied by Mashkani and Moslehi [2016]. They solve the problem with heuristic and branch-and-bound approaches.
When dealing with random failures, the stochastic case of the problem is considered. A survey of the integrated production-scheduling and maintenance-planning problem (IPSMP) with random failures is given by Hadidi et al. [2012a]. Cassady and Kutanoglu [2003] study the single-machine problem where the objective is to minimize the total weighted expected tardiness of all jobs. They use a full enumeration to solve the problem. In Cassady and Kutanoglu [2005], the objective is changed to the total weighted expected completion time. Sortrakul and Cassady [2007] propose a genetic algorithm for the problem with total weighted expected tardiness. Yulan et al. [2008] solve the multi-objective single-machine problem including five different objectives with a genetic algorithm. Hadidi et al. [2012b] use a mixed-integer non-linear formulation to solve the problem with a total-weighted-expected-completion-time objective. Wang [2013] studies a bi-objective version of the problem extending it with setup times. The problem is solved by using an evolutionary genetic algorithm. Wang and Liu [2013] propose a branch-and-bound algorithm for the single-machine problem where the objective is to minimize the total weighted expected completion time. An extension of the problem with imperfect preventive maintenance is laid out in the paper of Chen et al. [2015]. Abdelrahim and Vizvári [2017] study the single-machine problem to minimize the total expected completion time. They suggest a non-linear constrained 0-1-model which can be reduced to a non-linear unconstrained function by using problem-specific properties. Cui [2020] proposes an approximation proactive approach for the integrated problem with robust objectives.
There are two ways to formulate this scheduling problem, the rank and the general precedence formulation. At the first, jobs are assigned to ordered ranks to form the problem sequence. At the latter, the pairwise precedence comparison of each job represents the sequence. All of the stochastic references for a single machine have a rank formulation in common. To change the perspective, a single-machine problem where the objective is to minimize the total weighted expected completion time using a general precedence formulation in the modeling is examined in this work. This can be seen for the deterministic problem in the works of Chen [2006], Liu et al. [2016] and Krim et al. [2020]. With this different point of view, decision rules can be formulated to reduce the problem to a set-partitioning problem. This is solved with a branch-and-bound approach. The performance of the algorithm can be further improved by analyzing the constructed decision rules. This contributes to the capability of solving larger instances to optimality.
The paper is structured as follows. First, the problem is described in section 2. Properties of the problem and new solution features that result from extended decision rules are discussed in section 3. In section 4, the general branch-and-bound algorithm that uses the decision rules from the previous section in order to solve the problem is introduced. Lower and upper

bounds are also provided here. Further, two extended approaches to the general form are discussed. In section 5, the performances of the proposed approaches are evaluated and section 6 concludes the paper.

2 Problem description

There are n jobs j (or alternatively k) which need to be scheduled on a single machine. The machine is subjected to delays due to random failures. All jobs are available at time zero and have a given integer processing time P_j and a given weight W_j. The jobs should not interfere with each other. Hence, the period occupied in the time horizon starts with the expected starting time s_j and ends with the expected completion time c_j. The objective is to find the best sequence π that minimizes the total weighted expected completion time. The sequence is expressed by the variable x_{jk} that represents the predecessor-successor relationship between jobs j and k and forms the general precedence formulation of the problem.

$$x_{jk} = \begin{cases} 1, & \text{if job } j \text{ scheduled before job } k \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

The machine is assumed to be possibly unavailable due to uncertain corrective maintenance (CM) and planned preventive maintenance (PM). The CM is carried out because of random failures occurring while processing the jobs. PMs can be scheduled along the planning horizon at each beginning of a job being processed. This reduces the expected number of failures by reducing the machine's age. The age is defined as the total operation time of the machine since the last PM. During a CM, the machine is not available for the duration of T^{cm}, and T^{pm} during the PM, respectively. The probability of failures depends on the machine's age. This is depleted proportional to P_j and can be subdivided into the age prior (as_j) and post (ac_j) the processing of job j. It is assumed that the machine is minimally repaired at a CM. This means that this maintenance does not affect the age. With increasing age, the failure rate also increases which can be modeled by a Weibull distribution function with shape parameter $\beta > 1$ and scale parameter $\eta > 0$ [Cassady and Kutanoglu, 2003]. To control the increasing failure rate and thus the expected number of failures, PMs are needed. The age upon a PM is set back to an as-good-as-new state, called a perfect repair. Therefore, the stochastic deterioration process between two PMs can be seen as a non-homogeneous Poisson process. It counts the number of failures and renews with every PM. The interval between to consecutive PMs is further called a PM cycle. The expected number of failures during processing job j is

$$m(ac_j, as_j) = \left(\frac{ac_j}{\eta}\right)^{\beta} - \left(\frac{as_j}{\eta}\right)^{\beta} \tag{2}$$

as given in Cassady and Kutanoglu [2005].
With known duration of T^{cm} and T^{pm} and the above, expression the optimal PM cycle time τ^* can be calculated analytically [Cassady and Kutanoglu, 2003]. The maximization of the steady-state machine availability as a function of cycle length τ

$$A(\tau) = \frac{\tau}{\tau + m(\tau, 0)T^{cm} + T^{pm}} \tag{3}$$

leads with differentiation to

$$\tau^* = \eta \left[\frac{T^{pm}}{T^{cm}(\beta - 1)}\right]^{\frac{1}{\beta}}. \tag{4}$$

This cannot usually be met with the assumptions of integer processing times and a combinatorial approach is needed, balancing the delays resulting from PM and CM.

It is assumed that a PM is always scheduled prior to a job and the planning horizon starts with a PM. Hence, all PM cycles are identical as they start with a PM and jobs can be assigned to them using the binary variable

$$y_{jb} = \begin{cases} 1, & \text{if job } j \text{ is assigned to cycle } b \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where $b = 1, \ldots, n$ is the index of a PM cycle. The maximal number of all cycles b is equal to the number of all jobs j here because it is possible that each job can be assigned to its own cycle.
To define the expected starting and completion times of the different cycles, two additional variables ps_b and pc_b are used. An auxiliary variable that describes the accumulated processing times at the end of each cycle is called pa_b. This is needed to calculate the age within a cycle. Given this notation, the problem can be expressed as the following model (expressions (6) to (21)). Equation (7) is non-linear but can be linearized with a binary mapping from ages to expected number of failures. For a better overview, these constraints are omitted here.

$$\min \quad \sum_{j=1}^{n} W_j c_j \tag{6}$$

$$c_j = s_j + P_j + m(ac_j, as_j)T^{cm} \qquad \forall \quad j = 1, \ldots, n \tag{7}$$

$$s_k + M^c(1 - x_{jk}) \geq c_j \qquad \forall \quad j, k = 1, \ldots, n : j \neq k \tag{8}$$

$$s_j + M^c x_{jk} \geq c_k \qquad \forall \quad j, k = 1, \ldots, n : j \neq k \tag{9}$$

As mentioned above, the total weighted expected completion time should be minimized (6). Equation (7) expresses the expected completion time that depends on the expected starting time and the expected processing time. The latter consists of the deterministic part P_j and the stochastic part $m(ac_j, as_j)T^{cm}$. The expected number of failures depends on the decisions taken during the cycle assignment. Inequalities (8) and (9) ensure the non-interference of jobs j and k and determine the sequence. M^c is a sufficiently large number that is larger than the maximal possible expected completion time.

$$\sum_{b=1}^{n} y_{jb} = 1 \qquad \forall \quad j = 1, \ldots, n \tag{10}$$

$$c_j \leq pc_b + M^c(1 - y_{jb}) \qquad \forall \quad j, b = 1, \ldots, n \tag{11}$$

$$s_j \geq ps_b - M^c(1 - y_{jb}) \qquad \forall \quad j, b = 1, \ldots, n \tag{12}$$

$$ps_b \geq pc_{b-1} + T^{pm} \qquad \forall \quad b = 2, \ldots, n \tag{13}$$

Equation (10) ensures that every job is assigned to exactly one cycle. Inequalities (11) and (12) restrict s_j and c_j to the expected starting and completion times of the PM cycle they are assigned to. A PM is scheduled between two consecutive cycles (13). Here, a natural order of the cycles is assumed (which slightly differs from the later assumptions).

$$ac_j = as_j + P_j \qquad \forall \quad j = 1, \dots, n \tag{14}$$

$$as_j \geq -M^a(1 - y_{j1}) + \sum_{k=1}^{n} x_{kj} P_k \qquad \forall \quad j = 1, \dots, n \tag{15}$$

$$as_j \geq -pa_{b-1} - M^a(1 - y_{jb}) + \sum_{k=1}^{n} x_{kj} P_k \qquad \forall \quad j = 1, \dots, n, b = 2, \dots, n \tag{16}$$

$$pa_1 = \sum_{j=1}^{n} y_{j1} P_j \tag{17}$$

$$pa_b = pa_{b-1} + \sum_{j=1}^{n} y_{jb} P_j \qquad \forall \quad b = 2, \dots, n \tag{18}$$

Equation (14) represents the increasing age. Expressions (15) to (18) ensure the proper starting age of job j which depends on the assignment decision. Only the processing times of the predecessor jobs that belong to the same cycle are accumulated. This is the same as the combined processing times of all predecessors minus the accumulated times at the end of the previous cycle. A sufficiently large number M^a represents the maximal possible cycle age. Here, the sum of all processing times is used.
The last expressions ensure the start of the planning horizon and the domains of all decision variables.

$$ps_1 \geq T^{pm} \tag{19}$$

$$x_{jk}, y_{jb} \in \{0, 1\} \qquad \forall \quad j, k, b = 1, \dots, n \tag{20}$$

$$s_j, c_j, as_j, ac_j, ps_b, pc_b, pa_b \geq 0 \qquad \forall \quad j, b = 1, \dots, n \tag{21}$$

Preliminary tests show that the given formulation with a binary mapping is a possible representation of the described problem but can solve only instances up to 10 jobs. At this formulation, two decisions are needed. First, the precedence relationship of the jobs and second, the job assignment to PM cycles. Since the cycles are ordered in a natural way, the assignment to them is also a sequencing of its own. The problem can be also seen as one precedence relationship of the jobs, one of all cycles and an assignment to cycles without consideration of the ordering. In the next sections, some properties are discussed for the two precedence relationships and a branch-and-bound approach is presented for the assignment.

3 Some properties

Both decision rules discussed here are based on the well known Smith rule [Smith, 1956], also known as the Weighted Shortest Processing Time First (WSPT)-rule. The rule is optimal for scheduling on a single machine with total-weighted-completion-time objective [Pinedo, 2016]. By this rule, the jobs are sequenced in decreasing order of W_j/P_j. It is based on the so-called adjacent pairwise interchange of different jobs j and k as illustrated in figure 1.
The partial sequences before (π) and after (π') the two jobs are not affected by the adjacent pairwise interchange. The completion time of the second job is equal in both cases of the

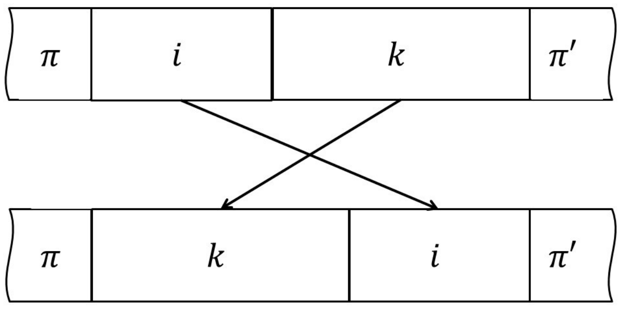

Figure 1: *Adjacent pairwise interchange*

interchange because it is the sum of both processing times. Hence, only the processing time of the first and the weights of both jobs determine the weighted completion time of the partial sequence [Pinedo, 2016].

3.1 Inner cycle dominance

To adapt the WSPT-rule to age-dependent processing times as given in this problem, Wang and Liu [2013] propose the inequality

$$\frac{W_j}{P_j + T^{cm}\left(\left(\frac{P_k+P_j+ac_\pi}{\eta}\right)^\beta - \left(\frac{P_k+ac_\pi}{\eta}\right)^\beta\right)} < \frac{W_k}{P_k + T^{cm}\left(\left(\frac{P_k+P_j+ac_\pi}{\eta}\right)^\beta - \left(\frac{P_j+ac_\pi}{\eta}\right)^\beta\right)} \tag{22}$$

to prove the dominance of a sequence where job j is scheduled before k over another sequence where k is scheduled before j at a given age ac_π of the partial sequence π. The rule can be used for every job assigned to the same cycle to obtain the inner cycle sequence. For this, the jobs are added to the partial sequence one by one. A job is selected by pairwise dominance comparison within every unscheduled job at a given age. After selecting a job, the age of the partial sequence is adjusted and another job is selected. This leads to the optimal sequence in a PM cycle [Wang and Liu, 2013].
The sequence dominance can also be calculated in a preprocessing phase. Both sides of the inequality (22) form a monotonically decreasing function for each age from zero to the maximal possible age. Therefore, the sequence of two jobs only changes at the intersections of these curves which can be calculated for each pair. Every predecessor-successor relationship can be evaluated by such a pairwise comparison. Logical induction can simplify this while processing. The optimal sequence decision can now be obtained by comparing the partial sequence age with these intersections. As an additional point, it can also be seen that for some weight-processing-time configurations of the jobs j and k, the dominance never changes in the considered range. Hence, if both are assigned to the same cycle the sequence decision stays the same and is not age-dependent. These jobs can be fixed which reduces the number of comparisons. The non-fixed sequences form different patterns in the fixed sequence which has to be recalculated to find the optimal sequence. The following example in figure 2 shows this behavior.
In this example, the jobs 1,5 and 6 are fixed due to the rule above indicating that predecessor-successor relationship are age-independent. Jobs 2 to 4 should be sequenced after job 1 and prior to 5 but the sequence is unknown and age-dependent. Same applies to job 7 and 8. The partial sequence of the first and the second group can be recalculated independently by sequencing one job at a time as given above.

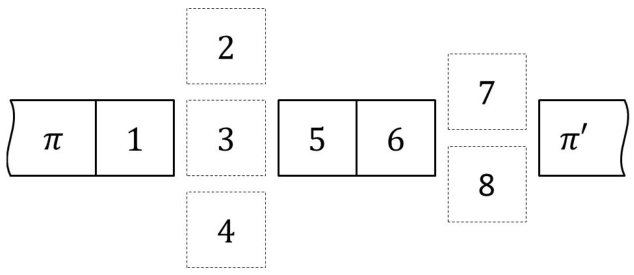

Figure 2: *Example for recalculation*

With this procedure the optimal sequence of a cycle where the age matters for the decision can be found. The determination of fixed jobs reduces the sequencing in the regular case and stays nearly the same in the worst-case where no jobs can be fixed.

3.2 Cycle dominance

The previous dominance rule determines the relationship of contiguous jobs assigned to the same cycle. A cycle can be seen as a partial sequence itself and two cycles can be sequenced according to an extension of the WSPT-rule.
A cycle with a given sequence of multiple jobs is a chain of jobs which have to be processed entirely before new jobs (or in this case cycles) can start. For the deterministic problem of scheduling on a single machine with precedence constraints and total-weighted-completion-time objective, the dominance rule

$$\frac{\sum_{j\in U} W_j}{\sum_{j\in U} P_j} > \frac{\sum_{k\in U'} W_k}{\sum_{k\in U'} P_k} \tag{23}$$

is given by Pinedo [2016]. Here U and U' represent two job sets of independent sequences. With this adjacent sequence interchange as a generalization of the adjacent pairwise interchange the sequence is given by sorting the chains in decreasing order of this ratio.
This rule can be adapted to the given problem. The sequence decision is age-independent because both chains start at an age of zero due to the PM scheduled at the beginning of the cycle. A schematic representation is shown in figure 3.

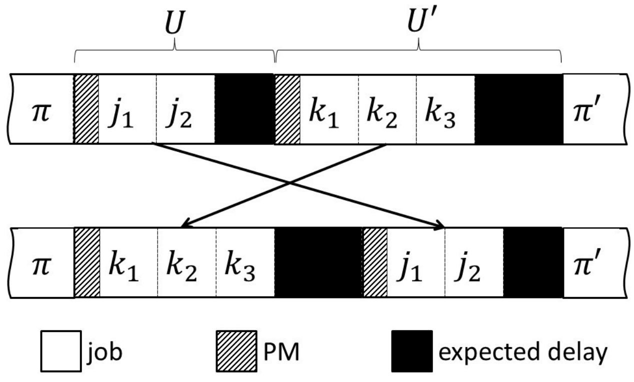

Figure 3: *Cycle dominance*

The numerators do not change in the inequality in comparison to (23). The denominators differ by exchanging the sum of processing times by the expected processing time (EP_j). It contains the sum of all processing times, the duration of a single PM (shaded parts) and the combined expected delays due to random failures (black parts). The delays depend on the age and therefore on the sum of processing times. This leads to the inequality

$$\frac{\sum_{j\in U} W_j}{T^{pm} + \sum_{j\in U} P_j + \left(\frac{\sum_{j\in U} P_j}{\eta}\right)^{\beta} T^{cm}} > \frac{\sum_{k\in U'} W_k}{T^{pm} + \sum_{k\in U'} P_k + \left(\frac{\sum_{k\in U'} P_k}{\eta}\right)^{\beta} T^{cm}}. \tag{24}$$

With inequalities (24) and (22), both decisions on the sequence inside the cycles and the sequence of the cycles can be tackled. Note that these rules and properties also pertain to other distributions which can reflect a monotonically increasing failure rate at increasing age. The remaining decision is on assigning jobs to disjunctive cycles for which a branch-and-bound algorithm can be used.

4 Solution approach

The branch-and-bound algorithm used in this work iterates over every possible cycle combination and uses lower and upper bounds on the problem to prune the search tree.
A node represents an assignment of jobs to a cycle and every level an additional cycle. Each assignment is feasible. The ordering of the assignment is irrelevant for the decision due to the cycle dominance rule. This decreases the search tree size. Furthermore, one job (here that one with the smallest index) can be directly assigned at every level because at each node at least one job has to be assigned. An illustration of the proposed search tree (left) and an example tree of a rank formulation (right) is shown in figure 4. The trees are only outlined and incomplete. The dashed arrows imply further nodes.

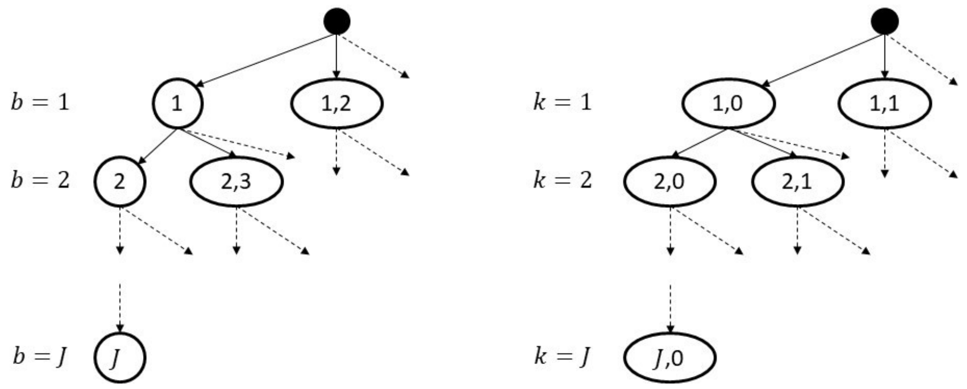

Figure 4: *Illustration of search tree structures*

For the proposed search tree on the left, the node labels represent the jobs assigned to this cycle. On the other tree, the first label stands for the job assigned to this rank. The second label indicates whether a PM is scheduled or not. At each rank every combination of unassigned jobs and PM decisions is possible.
The proposed search tree is smaller in size compared to the rank-based view of the problem due to the cycle dominance rule and therefore the reduction of the decision space. A depth-first search strategy is used here for tree traversal.

4.1 Lower bound

At every node of the search tree, a lower bound to the objective function value of the full solution can be obtained. To calculate this bound, dummy cycles are used for each

non-assigned job. These dummies represent a processing-time dependent portion of the analytically optimal cycle time τ^*. They are created once in the preprocessing phase. Then the dummies are sequenced with the current partial solution by the cycle dominance rule to calculate the total weighted expected completion time.
Assume a cycle with the sum of processing times equal to τ^*. The expected delays through CM and the planned delays through PM are now perfectly balanced and this results in an optimal expected completion time C^* of this cycle including T^{pm}, T^{cm} and τ^*. By scaling C^* proportionally to P_j/τ^*, a portion of C^* is created as a dummy cycle with the expected processing time

$$EP_j^{dummy} = \frac{P_j}{\tau^*}\left(T^{pm} + \tau^* + T^{cm}\left(\frac{\tau^*}{\eta}\right)^{\beta}\right) \tag{25}$$

and weight W_j. EP_j^{dummy} is always smaller than EP_j and therefore yields a lower bound for the objective function value of the full solution. This is due to the increase of processing times of the same number of jobs and the use of the optimal sequence in both cases also increases the total weighted expected completion time.
With this lower bound, the search tree can be pruned with every upper bound obtained.

4.2 Algorithm and extensions

The branch-and-bound algorithm (BB) is solved by traversing the whole tree depth-first utilizing the lower bound. Starting solutions like assigning the jobs in WSPT-order to the cycles and closing the cycles with respect to a myopic rule can be used [Wang and Liu, 2013] to obtain an upper bound, but were omitted to show the sole performance of the algorithm. However, the myopic rule is implemented in a heuristic approach that approximates the optimal solution. This second approach is called HBB. The search tree is here traversed as in BB. If the number of jobs in a cycle is greater than two, it is checked whether the last job of the obtained sequence should be assigned to this cycle or the next [Wang and Liu, 2013]. This is done by comparing the expected completion time of the job inside a cycle at given age ac_π with the resulting time in the next cycle with an age of zero and additional T^{pm}. Because the processing time is the same in both cases, it cancels out. Therefore, the inequality

$$T^{cm}\left(\left(\frac{ac_\pi + P_j}{\eta}\right)^{\beta} - \left(\frac{ac_\pi}{\eta}\right)^{\beta}\right) > T^{pm} + T^{cm}\left(\frac{P_j}{\eta}\right)^{\beta} \tag{26}$$

shows on the left-hand side the additional time if scheduled inside the cycle and on the right-hand side if the job is scheduled in the next cycle. If the inequality holds, the node is pruned. Every cycle assignment that contains this job combination and also further jobs can be treated the same. With this rule in the search procedure, near-optimal solutions can be found but it does not guarantee the optimal solution due to its myopic character. The idea of ordering the processing times in decreasing order to increase the impact of the myopic rule at HBB is omitted because preliminary tests did not show significant performance improvements.
As a third algorithm (HBB+BB), the upper bound obtained by HBB is used in a first phase and every node that is pruned due to the rule above is only expanded in a second phase. The nodes are now expanded to find the optimal solution and prune the tree better by utilization of the upper bound from the approximation.

5 Computational study

5.1 Data generation and design of experiments

The data generation is based on the computational study of Wang and Liu [2013]. Three groups of instances are generated which differ in the maintenance durations with values namely $(T^{pm}; T^{cm}) = (5; 10)$, $(50; 150)$ and $(120; 200)$ for groups 1 to 3. For every group, $10, 12, \ldots, 18$ jobs are generated with every combination of $\eta \in \{80; 100; 200\}$ and $\beta \in \{2; 3\}$. For every configuration, 30 random instances are generated by sampling P_j from a discrete uniform distribution $U[1; 100]$ and W_j from $U[1; 10]$. Hence, the data set includes a total of 2700 instances which can be found with all results in the supplementary material. Also some larger instances are tried, but preliminary tests indicate that instances with 20 jobs cannot be solved to proven optimality in the given time limit of 1800 seconds.
The three algorithms tested on this data set are HBB, BB and HBB+BB as described in the previous section. The preprocessing is calculated once for all approaches and is reported with the relative running time 'rt-pre' to the total running time 't-tot' (in seconds). The relative solution time of the algorithm 'rt-sol' is also given. If the algorithm terminates in the given time limit of 1800 seconds, it is marked as 'term'. The same applies for the best solution found 'best' if this algorithm finds the best or optimal solution over all algorithms. Since the HBB will not always find the optimal solution, the relative difference

$$\Delta = \frac{z_{HBB} - z^*}{z^*} \tag{27}$$

is also reported if the optimal solution is known. Here, z represents the objective function value. BB and HBB+BB are compared with a 'wins' measurement. An algorithm gets a point for every instance where it is faster than the other with a tolerance of 0.5% of the minimum of both running times. If the tolerance is not met, none of the algorithms gets a point. Only comparisons where both algorithms terminate are counted. All performance measurements are reported as the average over the 30 instances and only solutions where the algorithm terminates are included. The percentage of termination in the given time limit is also given.
The algorithms are implemented in Python and tests are run on an Intel(R) Core(TM) i5-6500 CPU machine at 3.2 GHz with 8 GB of RAM.

5.2 Results

The results can be seen in tables 1 to 3 for the different groups of T^{pm} and T^{cm} values.
It can be seen in the results that the performance differs between the three groups by investigating the 'term' measurement. The job configuration, where the algorithms cannot solve all instances, changes between the groups. In group 1, this begins at $n = 18$, in group 2 at $n = 16$, and in group 3 at $n = 14$. Because the other parameters stay the same, the results show that an increase in T^{pm} and T^{cm} decreases the performance of the algorithms. It is open how the different ratios of these parameters influence the performance. The number where HBB finds the optimal solutions drops over all groups from a maximum of 41% for the smaller instances to a minimum of 11% for the larger ones. However, the relative gap ranges from 0.4% to 2.3% with an average of 1.4% for all groups. It becomes apparent that the differences in groups seem to have a higher impact on the heuristic performance than the increase in job number. Inside the groups, there is just an increase in the relative gap

Table 1: *Results for group 1*

n	sol	term	best	rt-pre	rt-sol	t-tot	Δ	wins
	HBB	1	0.41	0.971	0.029	7.21	0.004	-
10	BB	1	1	0.911	0.089	7.62	-	0.94
	HBB+BB	1	1	0.902	0.098	7.69	-	0.06
	HBB	1	0.29	0.922	0.078	11.05	0.005	-
12	BB	1	1	0.728	0.272	13.73	-	0.82
	HBB+BB	1	1	0.713	0.287	14.02	-	0.18
	HBB	1	0.20	0.845	0.155	17.74	0.005	-
14	BB	1	1	0.408	0.592	37.02	-	0.85
	HBB+BB	1	1	0.390	0.610	39.13	-	0.15
	HBB	1	0.19	0.762	0.238	41.82	0.005	-
16	BB	1	1	0.148	0.852	171.04	-	0.77
	HBB+BB	1	1	0.142	0.858	173.86	-	0.23
	HBB	0.99	0.13	0.683	0.317	118.32	0.005	-
18	BB	0.87	1	0.039	0.961	771.81	-	0.62
	HBB+BB	0.84	1	0.040	0.960	755.04	-	0.36

ranging from 0.1 (group 1) to 0.6 (group 3) percent points. Comparing the averages of the groups, the gap increases from 0.4% to nearly 2% for groups 2 and 3.
The opposite trend in the relative times of preprocessing and solution time shows that the time spent for preprocessing increases less than the solution time. This is also reflected by the average preprocessing time which is nearly 18 seconds. It is obvious that HBB is faster than BB, because not the full tree needs to be traversed. BB and HBB+BB are always near comparing the runtime where BB is usually better than HBB+BB. This can also be seen from the instance wins where BB is always better. A reason for this can be the longer runtime of the HBB by first finding the last job and then checking the myopic rule. This can lead to a better upper bound but results show that the trade-off between longer runtime and runtime reduction through better pruning does not pay off for this instance size and runtime limit. It seems that group 3 shows a trend of both algorithms performing similarly. This could be the focus of further research.
Generally, this computational study shows that all tested algorithms are able to solve the largest instances in the test set even for groups 2 and 3. In comparison, Wang and Liu [2013] discuss that their proposed algorithm can solve up to 18 jobs only for group 1 and up to 14 jobs for group 2 and 3. However, the approaches proposed here differ in their focus. If a fast solution is prioritized over optimality, the HBB with the myopic rule can be used to find good near-optimal solutions with an approximately 2% gap. But if focusing more on optimality, the study shows that BB should be preferred over HBB+BB because the repeatedly checking of the above rule slows the solution process down. However, the better upper bound found by HBB+BB might improve the solution process when solving larger instances with a greater time limit.

Table 2: *Results for group 2*

n	sol	term	best	rt-pre	rt-sol	t-tot	Δ	wins
	HBB	1	0.34	0.962	0.038	9.79	0.016	-
10	BB	1	1	0.847	0.153	10.93	-	0.91
	HBB+BB	1	1	0.836	0.164	11.07	-	0.09
	HBB	1	0.22	0.902	0.098	15.49	0.021	-
12	BB	1	1	0.566	0.434	24.90	-	0.82
	HBB+BB	1	1	0.554	0.446	25.73	-	0.18
	HBB	1	0.13	0.828	0.172	27.90	0.020	-
14	BB	1	1	0.218	0.782	117.06	-	0.73
	HBB+BB	1	1	0.211	0.789	123.23	-	0.27
	HBB	1	0.11	0.765	0.235	58.95	0.019	-
16	BB	0.89	1	0.056	0.944	591.42	-	0.63
	HBB+BB	0.90	1	0.056	0.944	616.86	-	0.36
	HBB	0.97	0.14	0.619	0.381	144.77	0.018	-
18	BB	0.18	1	0.039	0.961	978.80	-	0.84
	HBB+BB	0.19	1	0.035	0.965	1058.14	-	0.12

6 Conclusion

In this paper, a branch-and-bound approach and variations of this baseline algorithm are described for solving the IPSMP for a single machine with random failures. The objective is to schedule weighted jobs and preventive maintenance activities to reduce the impact of random failures and minimize the total weighted expected completion time. The probability of failures depends on the machine's age which is controlled through the maintenance planning. Several approaches are discussed in the literature to address this problem. Common to all is the use of a rank formulation of the problem. The general precedence formulation used in this work gives rise to an extension of the decision rules which reduces the problem to a set-partitioning problem. A mixed-integer formulation is given that is capable of solving small-size problems. Further, ideas are discussed to reduce the computational effort in the solution phase with a preprocessing using problem-specific properties. The set-partitioning problem is solved by use of a branch-and-bound algorithm and a developed lower bound. To find near-optimal solutions fast, a heuristic version of the exact branch-and-bound algorithm with a myopic rule can be used instead of the normal algorithm. Also a setting of combining both versions is tested but cannot outperform the others. However, the results show that the branch-and-bound procedure developed in this work is well suited to solve problems up to 18 jobs and therefore shows better performance compared to Wang and Liu [2013]. It could be interesting to see how further upper and lower bound approaches, initial sorting of jobs and other traverse strategies can increase the performance on even larger instances.

Table 3: *Results for group 3*

n	sol	term	best	rt-pre	rt-sol	t-tot	Δ	wins
10	HBB	1	0.39	0.917	0.083	9.47	0.019	-
	BB	1	1	0.816	0.184	10.59	-	0.92
	HBB+BB	1	1	0.801	0.199	10.81	-	0.08
12	HBB	1	0.27	0.795	0.205	18.02	0.020	-
	BB	1	1	0.522	0.478	26.35	-	0.88
	HBB+BB	1	1	0.504	0.496	28.15	-	0.12
14	HBB	1	0.24	0.687	0.313	67.62	0.017	-
	BB	0.99	1	0.202	0.798	152.25	-	0.75
	HBB+BB	0.99	1	0.198	0.802	165.71	-	0.25
16	HBB	0.94	0.20	0.670	0.330	195.53	0.020	-
	BB	0.78	1	0.064	0.936	548.10	-	0.62
	HBB+BB	0.76	1	0.066	0.934	539.13	-	0.36
18	HBB	0.72	0.12	0.739	0.261	128.02	0.023	-
	BB	0.31	1	0.026	0.974	1148.34	-	0.53
	HBB+BB	0.30	1	0.025	0.975	1120.83	-	0.43

References

Abdelrahim, E. H. and Vizvári, B. [2017]. Simultaneous Scheduling of Production and Preventive Maintenance on a Single Machine, *Arabian Journal for Science and Engineering* **42**(7): 2867–2883.

Batun, S. and Azizoğlu, M. [2009]. Single machine scheduling with preventive maintenances, *International Journal of Production Research* **47**(7): 1753–1771.

Bock, S., Briskorn, D. and Horbach, A. [2012]. Scheduling flexible maintenance activities subject to job-dependent machine deterioration, *Journal of Scheduling* **15**(5): 565–578.

Cassady, C. R. and Kutanoglu, E. [2003]. Minimizing Job Tardiness Using Integrated Preventive Maintenance Planning and Production Scheduling, *IIE Transactions* **35**(6): 503–513.

Cassady, C. R. and Kutanoglu, E. [2005]. Integrating Preventive Maintenance Planning and Production Scheduling for a Single Machine, *IEEE Transactions on Reliability* **54**(2): 304–309.

Chen, J.-S. [2006]. Using integer programming to solve the machine scheduling problem with a flexible maintenance activity, *Journal of Statistics and Management Systems* **9**(1): 87–104.

Chen, X., Xiao, L. and Zhang, X. [2015]. A production scheduling problem considering random failure and imperfect preventive maintenance, *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* **229**(1): 26–35.

Cui, W. [2020]. Approximate Approach to Deal with the Uncertainty in Integrated Production Scheduling and Maintenance Planning, *Journal of Shanghai Jiaotong University (Science)* **25**(1): 106–117.

Hadidi, L. A., Turki, U. M. A. and Rahim, A. [2012a]. Integrated models in production planning and scheduling, maintenance and quality: a review, *International Journal of Industrial and Systems Engineering* **10**(1): 21.

Hadidi, L. A., Turki, U. M. A. and Rahim, M. A. [2012b]. Joint job scheduling and preventive maintenance on a single machine, *International Journal of Operational Research* **13**(2): 174.

Kim, B. S. and Ozturkoglu, Y. [2013]. Scheduling a single machine with multiple preventive maintenance activities and position-based deteriorations using genetic algorithms, *The International Journal of Advanced Manufacturing Technology* **67**(5-8): 1127–1137.

Krim, H., Benmansour, R., Duvivier, D., Aït-Kadi, D. and Hanafi, S. [2020]. Heuristics for the single machine weighted sum of completion times scheduling problem with periodic maintenance, *Computational Optimization and Applications* **75**(1): 291–320.

Lee, C.-Y., Lei, L. and Pinedo, M. [1997]. Current trends in deterministic scheduling, *Annals of Operations Research* **70**: 1–41.

Liu, M., Wang, S., Chu, C. and Chu, F. [2016]. An improved exact algorithm for single-machine scheduling to minimise the number of tardy jobs with periodic maintenance, *International Journal of Production Research* **54**(12): 3591–3602.

Low, C., Ji, M., Hsu, C.-J. and Su, C.-T. [2010]. Minimizing the makespan in a single machine scheduling problems with flexible and periodic maintenance, *Applied Mathematical Modelling* **34**(2): 334–342.

Ma, Y., Chu, C. and Zuo, C. [2010]. A survey of scheduling with deterministic machine availability constraints, *Computers & Industrial Engineering* **58**(2): 199–211.

Mashkani, O. and Moslehi, G. [2016]. Minimising the total completion time in a single machine scheduling problem under bimodal flexible periodic availability constraints, *International Journal of Computer Integrated Manufacturing* **29**(3): 323–341.

Pinedo, M. L. [2016]. *Scheduling*, Springer International Publishing, Cham.

Qi, X., Chen, T. and Tu, F. [1999]. Scheduling the Maintenance on a Single Machine, *The Journal of the Operational Research Society* **50**(10): 1071.

Sadiqi, A., El Abbassi, I., El Barkany, A. and El Biyaali, A. [2018]. Joint Scheduling of Jobs and Variable Maintenance Activities in the Flowshop Sequencing Problems: Review, Classification and Opportunities, *International Journal of Engineering Research in Africa* **39**: 170–190.

Sanlaville, E. and Schmidt, G. [1998]. Machine scheduling with availability constraints, *Acta Informatica* **35**(9): 795–811.

Schmidt, G. [2000]. Scheduling with limited machine availability, *European Journal of Operational Research* **121**(1): 1–15.

Smith, W. E. [1956]. Various optimizers for single-stage production, *Naval Research Logistics Quarterly* **3**(1-2): 59–66.

Sortrakul, N. and Cassady, C. R. [2007]. Genetic algorithms for total weighted expected tardiness integrated preventive maintenance planning and production scheduling for a single machine, *Journal of Quality in Maintenance Engineering* **13**(1): 49–61.

Wang, S. [2013]. Bi-objective optimisation for integrated scheduling of single machine with setup times and preventive maintenance planning, *International Journal of Production Research* **51**(12): 3719–3733.

Wang, S. and Liu, M. [2013]. A branch and bound algorithm for single-machine production scheduling integrated with preventive maintenance planning, *International Journal of Production Research* **51**(3): 847–868.

Xu, D., Liu, A. and Yang, D.-L. [2014]. Mathematical Programming Models for Competitive Two-Agent Single-Machine Scheduling with Flexible Periodic Maintenance Activities, *Arabian Journal for Science and Engineering* **39**(5): 3715–3722.

Xu, D. and Yin, Y. [2011]. On single-machine scheduling with flexible maintenance activities, *The International Journal of Advanced Manufacturing Technology* **56**(9-12): 1139–1145.

Yang, S.-J. and Yang, D.-L. [2010]. Minimizing the total completion time in single-machine scheduling with aging/deteriorating effects and deteriorating maintenance activities, *Computers & Mathematics with Applications* **60**(7): 2161–2169.

Yulan, J., Zuhua, J. and Wenrui, H. [2008]. Multi-objective integrated optimization research on preventive maintenance planning and production scheduling for a single machine, *The International Journal of Advanced Manufacturing Technology* **39**(9-10): 954–964.

Zhao, C.-L. and Tang, H.-y. [2010]. Single machine scheduling with general job-dependent aging effect and maintenance activities to minimize makespan, *Applied Mathematical Modelling* **34**(3): 837–841.

Chapter 4

Article 3: Decomposition approach for integrated production and maintenance scheduling on parallel machines

Decomposition approach for integrated production scheduling and maintenance planning on parallel machines

Sven Pries, Celso Gustavo Stall Sikora

Abstract

In modern industrial manufacturing, the high availability of production capacities is as important as it has never been before. To preserve it and hedge against delays due to random failure, preventive maintenance activities are needed. On the one hand, they restore the machine's condition but on the other hand they block it for production. Thus, the maintenance times are traded off against the delays caused by failures. This is incorporated into an identical parallel-machine scheduling problem minimizing the maximal expected completion time. The problem presented can be decomposed in multiple ways to accelerate the solution process. At every individual machine, the scheduling of jobs and preventive maintenance activities can be efficiently solved using a Dantzig-Wolfe decomposition and a branch-and-price algorithm. For the assignment of jobs to machines, a Benders' decomposition is used to utilize the lower bound information of the branch-and-price algorithm and combine it with combinatorial cuts. Therefore, the developed approach integrates a branch-and-price algorithm as subproblem of a Benders' decomposition. Computational results as well as further ideas to improve the performance are discussed and the proposed approach shows good solving capabilities compared to the monolithic model.

Keywords: scheduling, parallel machines, random failures, Benders, Dantzig-Wolfe

1 Introduction

In the scheduling literature, machines are usually assumed to be continuously available which might not be the case for a real industrial setting [Ma et al., 2010]. In realistic circumstances, machines may be non-available due to (lunch) breaks, planned preventive maintenance or corrective maintenance caused by random failures. Hence, schedules which incorporate the machine availability or the presence of delays through failures are more suitable for this kind of setting. While the single-machine problem can be seen as the building block for more complex systems [Schmidt, 2000], a parallel-machine environment might be more of interest for an error-prone system. First, parallel machines are very common in modern production industry [Li et al., 2017] and second, the non-availability or maintenance of one machine can be coped with by a parallel one at the same production stage.
In the deterministic version of the problem, the so-called scheduling with availability constraints, two cases can be distinguished, namely the fixed and the flexible case. For the fixed case, all intervals when the machine is not available for production are known in advance and cannot be changed. Comprehensive reviews for this case can be found in Schmidt [2000], Ma et al. [2010] and Kaabi and Harrath [2014].

If the flexible case is considered, the intervals are subjected to a decision. This can affect the duration, the number and/or the starting time depending on the problem. Partial reviews can be found in Hadidi et al. [2012a] and Sadiqi et al. [2018]. Different ways to introduce this to the parallel-machine problem are given in the literature. Sometimes, it is assumed that the machines have to be maintained once in the planning horizon while the starting time is fully flexible as done in Lee and Chen [2000] and Rebai et al. [2013]. Others assume that the starting and/or end times should be in a given time window as assumed in the works of Chen [2006a,b], Huo and Zhao [2011] and Liao et al. [2017]. In Sun and Li [2010], the time intervals between two consecutive non-availability periods must be smaller than a given value. In the works of Xu and Yang [2013] and Li et al. [2017], intervals are treated as jobs that need to be scheduled. An extension introduces deterioration to these maintenance jobs [Yang, 2013; Wang et al., 2014].
If random failures are considered, the stochastic case of this integrated problem is observed. For this, a comprehensive review can also be found in Hadidi et al. [2012a]. Cassady and Kutanoglu [2003] are the first considering the combined scheduling of production jobs and maintenance activities to reduce the delays caused by random failures. Here, the total expected weighted tardiness is minimized at a single-machine environment. They model the problem with an age-dependent non-homogeneous Poisson (NHP) process with the assumption of minimal repair. At failure, the machine is set back here to an operational state without changing the machine's age. This way of considering the problem is discussed in more detail in section 2. Cassady and Kutanoglu [2005] extend the problem to the total weighted expected completion time and solve it with full enumeration for small instances and a heuristic for larger ones. Sortrakul and Cassady [2007] propose a genetic algorithm for the single-machine problem with the objective of total weighted expected tardiness. Yulan et al. [2008] study a multi-objective single-machine problem with five objectives solving it with a special genetic algorithm. Hadidi et al. [2012b] solve the single-machine problem as a mixed-integer non-linear program where the objective is to minimize the total weighted expected completion time. In Wang [2013], a bi-objective (total expected completion time and the maximum of expected failure times) single-machine problem is solved with an evolutionary genetic algorithm. Wang and Liu [2013] propose a branch-and-bound approach for a single machine with minimization of the total weighted expected completion time. Wang and Liu [2016] solve a flowshop problem minimizing the maximal expected completion time with a genetic algorithm. Abdelrahim and Vizvári [2017] give a non-linear model for the minimization of the total expected completion time for a single machine. They show that this can be reduced to an unconstrained 0-1 optimization problem and can be solved by an algorithm using problem-specific properties. Cui et al. [2018] consider the flowshop environment and use a bi-objective of quality and solution robustness. In addition to the decisions on sequence and maintenance position, idle time can be inserted into the schedule. A two-loop algorithm is used to solve this problem. Cui [2020] gives an approximation approach for the single-machine problem minimizing the weighted sum of quality and solution robustness. A three-stage algorithm incorporating a gradient-descent algorithm using an effective surrogate measure is developed to solve the problem.
Other approaches to model the failure process can be also found. Mokhtari et al. [2012] study a reliability/availability-based approach for a parallel-machine setting. They use an exponential distribution function to model the failure and repair rates of the system. A population-based variable neighborhood search is used to solve this problem. Bajestani et al. [2014] study a flowshop environment minimizing the expected costs. The machine degradation is modeled using a continuous time homogeneous markov chain. They decompose the

problem and propose a combination of markov decision process and mixed-integer program to solve it. In von Hoyningen-Huene and Kiesmüller [2015], the expected makespan on parallel machines subjected to random failures is evaluated. In contrast to the NHP modeling approach, no minimal repair at failure is assumed. They propose two approximations for the obtained exact formula. Seif et al. [2019] study a flowshop problem minimizing the total expected cost. They used a two-stage stochastic mixed-integer program for small instances and a simulation-optimization with a genetic algorithm for larger ones. The degradation is modeled using health states with scenario-based transitions. A parallel-machine scheduling problem with periodic maintenance under a normal uncertainty distribution is investigated by Shen and Zhu [2019]. They show the worst-case bound for the longest-processing-time-first rule and propose an improved version.
An overview and classification of the literature for the stochastic case can be found in table 1.

Table 1: *Literature overview for the integrated stochastic case*

	modeling		machine environment			objective		approach			
Author(s) (Year)	NHP	other	single	parallel	flowshop	single	multi	meta-heuristic	math. programming	decomposition	other
Cassady and Kutanoglu [2003]	•		•			•					•
Cassady and Kutanoglu [2005]	•		•			•					•
Sortrakul and Cassady [2007]	•		•			•		•			
Hadidi et al. [2012b]	•		•			•			•		
Yulan et al. [2008]	•		•				•	•			
Mokhtari et al. [2012]		•		•		•		•			
Wang [2013]	•		•				•	•			
Wang and Liu [2013]	•		•			•					•
Bajestani et al. [2014]		•			•	•			•	•	
von Hoyningen-Huene and Kiesmüller [2015]		•		•		•					•
Wang and Liu [2016]	•				•	•		•			
Abdelrahim and Vizvári [2017]	•		•			•					•
Cui et al. [2018]	•				•		•	•			
Seif et al. [2019]		•			•	•		•	•		
Shen and Zhu [2019]		•		•		•		•			
Cui [2020]	•		•				•	•			
This paper	•			•		•			•	•	

From the literature review given in this paper, it becomes apparent that much work is done for the parallel-machine environment in the deterministic case. However, less work is done for the stochastic case (the papers by von Hoyningen-Huene and Kiesmüller [2015] and Shen and Zhu [2019]). Further, mixed-integer programming approaches (linear and non-linear) are only proposed by Hadidi et al. [2012b], Bajestani et al. [2014] and Seif et al. [2019]. Even fewer decomposition approaches are discussed in the literature (only by Bajestani et al. [2014]). In this paper, a mixed-integer quadratic program (MIQP) is developed to solve the identical parallel-machine problem with random failures. To accelerate the solution process, the problem is decomposed. The assignment of jobs to machines and the calculation of the maximal expected completion time can be solved separately in a Benders' decomposition framework. The resulting Benders' approach is further decomposed using a Dantzig-Wolfe decomposition. It is shown how the proposed decomposition transforms the difficult non-linear constraints to a well-manageable quadratic objective function in the Benders' subproblems. The additional decomposition leaves a strong linear relaxation in

the Benders' subproblem and enables the better use of classical Benders' cuts. Further, it preserves the quadratic objective function in the subproblem of the used branch-and-price algorithm.
The rest of this work is structured as follows. First, the problem is described in section 2. The solution approach decomposing the problem and enabling various cuts is discussed in section 3. The monolithic model and different variants of the proposed decomposition are evaluated in a computational study in section 4. In section 5, the work and results are concluded.

2 Problem description

The problem discussed in this paper is an integrated production-scheduling and maintenance-planning problem (IPSMP) on identical parallel machines. The objective is to minimize the maximal expected completion time and the machines are subjected to random failures. A set of jobs J has to be assigned to a set of identical machines I. The processing times of the jobs are assignment-independent and the same for all machines. The production of jobs increases the machine's age which is then more likely to fail. The age is defined as the total operational time of the machine since the last planned maintenance activity. At failure, a corrective maintenance (CM) of the machine is necessary which delays the completion of the current job. To reduce the expected delays of the job completion, preventive maintenance (PM) activities can be scheduled in the planning horizon at every individual machine. These activities reset the machine's age but block it for the processing of jobs. The time interval between two consecutive maintenance activities can be seen as a PM cycle. Therefore, the jobs have to be assigned to the elements of a set of PM cycles K on the machines to trade-off the delays resulting from random failures and scheduled PM activities. Because every job has to be assigned to one cycle, there can be as many cycles as jobs. Hence, $|J| = |K|$. A descriptive example of the problem is given in figure 1.

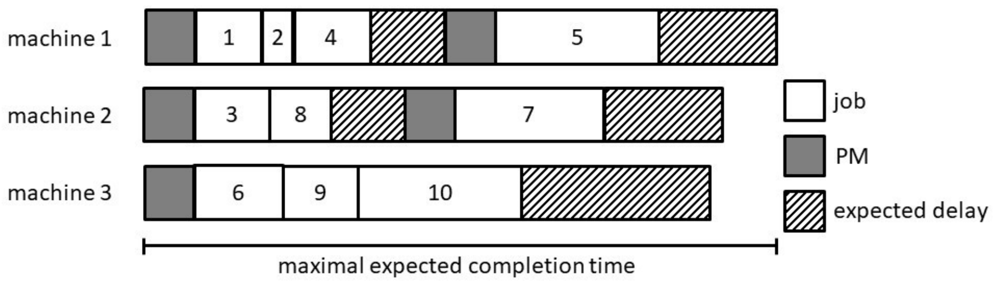

Figure 1: *Descriptive example*

Here, ten jobs need to be assigned to three machines. The PMs are represented by the dark squares. It is assumed that every cycle starts with a PM and ends before the next. Therefore, every machine is maintained at the beginning of the planning horizon. The maximal expected completion time of the schedule is given by the maximal expected completion time of the individual machines. The expected completion time of a machine is composed of the completion times of the PM cycles. It is assumed that the number of expected failures is sequence-independent. Hence, the expected delays within a cycle can be combined as given by the shaded rectangles. The delays of each cycle depend on the age of the machine and therefore on the sum of all processing times assigned to it.
Upon completion of a PM activity, the machine is perfectly repaired which means that the machine's age is set back to zero. The machine is then blocked for the duration of T^{pm}.

At every failure during a PM cycle, the machine undergoes a corrective maintenance with duration T^{cm}. Here, the machine is assumed to be minimally repaired, meaning that the machine is set back to an operational state without changing its age.
Knowing the age at the beginning and the end of a PM cycle, the cycle can be described as a non-homogeneous Poisson process [Cassady and Kutanoglu, 2003]. A Weibull hazard function

$$z(t) = \frac{\beta}{\eta^\beta} t^{\beta-1} \tag{1}$$

with the scale parameter $\eta > 0$ and shape parameter $\beta > 1$ is assumed, where the latter ensures an increasing failure rate. To obtain a well-manageable quadratic problem, $\beta = 2$ is assumed later in this work but the generalized form is given for completeness. Because identical machines are assumed, η, β and the maintenance durations (T^{pm} and T^{cm}) are the same for all machines.
Given (1) and end age a_{ik} of cycle k on machine i, the expected number of failures

$$\xi_{ik} = \left(\frac{a_{ik}}{\eta}\right)^\beta = \int_0^{a_{ik}} z(t)dt \tag{2}$$

can be calculated [Cassady and Kutanoglu, 2005]. The end age of a cycle k on machine i is defined by the sum of all processing times assigned to it and can be expressed as

$$a_{ik} = \sum_{j \in J} P_j x_{ijk} \tag{3}$$

with the job processing time P_j and the binary assignment variable

$$x_{ijk} = \begin{cases} 1, & \text{if job } j \text{ is assigned to cycle } k \text{ on machine } i \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The expected completion time of an individual machine i is given by c_i. The maximum of these expected completion times is the maximal expected completion time of the schedule $c_{\max}$. Note that the expected-value problem and no stochastic programming approach is used here. Both can differ in solution due to the maximization over machines. If jobs are assigned to a PM cycle, the binary cycle choice variable cc_{ik} is 1 if jobs are assigned to cycle k on machine i, and 0 otherwise.
With differentiation and algebraic analysis, the optimal time between two consecutive PMs (and therefore the optimal age)

$$a^* = \eta \left[\frac{T^{pm}}{T^{cm}(\beta - 1)}\right]^{\frac{1}{\beta}} \tag{5}$$

is obtained. With the assumption of integer processing times and non-resumable or splitable jobs, this cannot be met in every cycle. Therefore, a combinatorial approach is needed to trade-off the delays. However, a^* can be used to form a lower bound on the expected completion time of a machine as discussed further in section 3.
With the summarized notation in table 2, a monolithic formulation of the problem is given by

$$\min c_{\max} \tag{6}$$

$$c_{\max} \geq c_i \qquad \forall \quad i \in I \tag{7}$$

Table 2: *Problem notations*

description	domain	type	meaning
I		index	set of machines i
J		index	set of jobs j
K		index	set of cycles k
P_j	(J)	data	processing time of job j
T^{pm}		data	duration of PM
T^{cm}		data	duration of CM
η		data	scale parameter of Weibull hazard function
β		data	shape parameter of Weibull hazard function
x_{ijk}	(I,J,K)	binary	assignment of job j to PM cycle k on machine i
cc_{ik}	(I,K)	binary	indicates if PM cycle k on machine i is used for scheduling
c_i	(I)	$\mathbb{R}^+$	expected completion time of machine i
$c_{\max}$		$\mathbb{R}^+$	maximal expected completion time of the machines

$$c_i = \sum_{k\in K}\left(cc_k T^{pm} + \sum_{j\in J} P_j x_{ijk} + T^{cm}\left(\frac{\sum_{j\in J} P_j x_{ijk}}{\eta}\right)^{\beta}\right) \qquad \forall \quad i \in I \tag{8}$$

$$\sum_{i\in I,k\in K} x_{ijk} = 1 \qquad \forall \quad j \in J \tag{9}$$

$$x_{ijk} \le cc_{ik} \qquad \forall \quad i \in I, j \in J, k \in K \tag{10}$$

$$c_i \ge c_{i+1} \qquad \forall \quad i \in I : i \ne |I| \tag{11}$$

$$cc_{ik} \ge cc_{ik+1} \qquad \forall \quad i \in I, k \in K : k \ne |K| \tag{12}$$

$$c_{\max}, c_i \ge 0 \qquad \forall \quad i \in I \tag{13}$$

$$x_{ijk}, cc_i \in \{0,1\} \qquad \forall \quad i \in I, j \in J, k \in K. \tag{14}$$

Expressions (6) and (7) ensure that the maximum of the expected completion times over all machines is minimized. With equation (8), the expected completion time of an individual machine is calculated, which is the sum of the expected processing times of all cycles. The first term represents the fixed PM duration that is necessary if jobs are assigned to this cycle. The second term is the sum of all processing times assigned. The last part expresses the accumulated expected delays due to random failure where ξ_{ik} is reformulated using $\sum_{j\in J} P_j x_{ijk}$ instead of a_{ik}. Each job has to be assigned to exactly one cycle on one machine (9). Inequality (10) ensures that the cycle choice variable must be one if a job is assigned to this cycle. Because of the identical machines and the identical PM cycles at every machine, a high symmetry is present in the model. Therefore, two symmetry breaking constraints are used. Inequality (11) ensures that the machines are ordered by the completion times in descending order. A similar order is ensured by (12) for the PM cycles on each machine. (13) and (14) define the domains given in table 2.

It can be seen that due to equation (8), the monolithic model contains non-linear constraints. To be able to solve this model as a mixed-integer program (MIP), a binary mapping with the mapping variable θ_{ikb} from a_{ik} to ξ_{ik} can be used, as is done in this work. Every possible age A_b (derived from the combination of all processing times) and the corresponding expected number of failure F_b are given to the model as data with the mapping set B. Therefore, expression (8) is modified as

$$c_i = \sum_{k \in K} \left(cc_k T^{pm} + \sum_{j \in J} P_j x_{ijk} + T^{cm} \sum_{b \in B} F_b \theta_{ikb} \right) \qquad \forall \quad i \in I \tag{15}$$

with additional constraints

$$\sum_{j \in J} P_j x_{ijk} = \sum_{b \in B} A_b \theta_{ikb} \qquad \forall \quad i \in I, k \in K \tag{16}$$

$$\sum_{b \in B} \theta_{ikb} = 1 \qquad \forall \quad i \in I, k \in K \tag{17}$$

$$\theta_{ikb} \in \{0, 1\} \qquad \forall \quad i \in I, k \in K, b \in B \tag{18}$$

where (16) and (17) ensure that one of the possible ages is mapped to the cycle age. In the last term of the modified equation, the mapping variable is used to express the corresponding expected number of failures.

3 Solution approaches

3.1 Decomposition idea

To accelerate the solution process, the monolithic model is decomposed into smaller problems using Benders' and Dantzig-Wolfe decompositions.
The Benders' decomposition was first proposed by Benders [1962] and is based on the division of the formulation into two sets: a master problem (MP) and one or more subproblems (SPs). The division can be advantageous when the partial problems can be solved faster than the whole formulation without the decomposition. The MP is formulated in a way, that its solution represents a lower bound (for minimization problems) of the problem. After each solution of the MP, the SPs can be solved to obtain the actual objective value of the obtained solution. This objective value is then given back to the MP via valid inequalities [Rahmaniani et al., 2017]. The classical version of the algorithm [Benders, 1962] considers linear SPs and uses shadow prices to formulate the valid inequalities. Formulations containing integer variables in the SP can also be decomposed in this framework using combinatorial cuts [Laporte and Louveaux, 1993].
Variable x_{ijk} from the monolithic model describes the assignment of a job to a PM cycle at a machine. This decision can be decomposed into the assignment of jobs to machines and the assignment of jobs to PM cycles for every individual machine. In this approach, a Benders' decomposition is used to divide the decisions in a MIP master problem for the first assignment and a MIQP subproblem for the latter. The formulations of both problems and the valid inequalities (or cuts) used to transfer information between the problems are discussed in detail in the following section.
If the first assignment is given by the MP, a single-machine assignment problem of jobs to PM cycles remains for each machine. Every PM cycle assignment can be seen as an individual

problem. The collection of all individual problems has to cover all the jobs assigned to an individual machine. This second problem can be reformulated again using a Dantzig-Wolfe decomposition. The problem is decomposed into a linear restricted master problem (RMP) in the form of a set-partitioning problem. Each column represents a feasible assignment of jobs to a single PM cycle and not all possible columns are present from the beginning. Much rather, promising columns are generated in an iterative process, the column generation. For a primer to column generation, see Desrosiers and Lübbecke [2005]. Here, new columns are found by solving a MIQP subproblem. The negative reduced costs of a potential column are minimized given the dual variable values from the RMP. This is then added to the master problem until no further columns can be found. The linear RMP can converge to a non-integral solution. To ensure integrality, a branching procedure is used then. This branch-and-price approach finds the optimal integral solution and is used for each individual machine.

3.2 Benders' decomposition

The monolithic model is decomposed into a MP for the assignment of jobs to machines and a SP for the assignment of jobs to PM cycles for every individual machine. The MP generates solutions by approximating the resulting maximal expected completion time of the assignment which is a lower bound for the actual value. For every machine, the SP is solved and the real value is obtained. The information about the actual values of the assignment is integrated into the MP using cuts that make the approximation of the MP more precise. A lower bound using the analytical optimal PM cycle length (5) is used to better approximate the resulting expected completion time of every machine and therefore the maximum of them. At a cycle length of a^*, the delays through PM and CM are perfectly balanced. The total assigned processing time to a machine can be seen as a fraction of needed PM cycles by dividing it by a^*. For these analytically optimal PM cycles, the expected delays from PM and CM can be calculated.
For the formulation of the Benders' MP and SP, the notation of table 2 is adjusted by replacing the binary assignment variable x_{ijk} by the variables x_{ij}, for the assignment of the MP, and y_{jk}, for the assignment of the SP. The MP can then be formulated as the following MIP.

$$\min \quad c_{\max} \tag{19}$$

$$c_{\max} \geq c_i \qquad \forall \quad i \in I \tag{20}$$

$$c_i = \left(\sum_{j \in J} P_j x_{ij}\right)\left(1 + \frac{T^{pm} + T^{cm}\left(\frac{a^*}{\eta}\right)^{\beta}}{a^*}\right) \qquad \forall \quad i \in I \tag{21}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall \quad j \in J \tag{22}$$

$$c_i \geq c_{i+1} \qquad \forall \quad i \in I : i \neq |I| \tag{23}$$

$$c_{\max}, c_i \geq 0 \qquad \forall \quad i \in I \tag{24}$$

$$x_{ij} \in \{0,1\} \qquad \forall \quad i \in I, j \in J \tag{25}$$

Here, expressions (19), (20) and (23) are the same as in the monolithic model. Equation (21) represents the discussed lower bound to the machine assignment. The modified equation (22) ensures that every job is assigned to one machine.

The MP generates a machine assignment and for every individual machine, the following subproblem is solved to obtain the actual maximal expected completion time after the PM cycle assignment. Just a subset of jobs J is assigned at the subproblem given by the MP. This subset is defined as $\hat{J}$.

$$\min \quad \sum_{k\in K}\left(cc_k T^{pm} + \sum_{j\in\hat{J}} P_j y_{jk} + T^{cm}\left(\frac{\sum_{j\in\hat{J}} P_j y_{jk}}{\eta}\right)^{\beta}\right) \tag{26}$$

$$\sum_{k\in K} y_{jk} = 1 \qquad \forall \quad j\in\hat{J} \tag{27}$$

$$y_{jk} \le cc_k \qquad \forall \quad j\in\hat{J}, k\in K \tag{28}$$

$$cc_k \ge cc_{k+1} \qquad \forall \quad k\in K : k \ne |K| \tag{29}$$

$$y_{jk}, cc_k \in \{0,1\} \qquad \forall \quad j\in\hat{J}, k\in K \tag{30}$$

For an individual machine, the objective (26) is to minimize the expected completion time which is the same as (8). Equation (27) ensures that each assigned job in $\hat{J}$ is assigned to exactly one PM cycle. Expressions (28) and (29) are the same as in the monolithic model without the machine index.

For each solution of the subproblem with objective function value c_i^{sub}, a combinatorial Benders' cut (CC)

$$c_{\max} \ge \left(\sum_{j\in\hat{J}_i} x_{ij} - |\hat{J}_i| + 1\right) c_i^{sub} \qquad \forall \quad i\in I \tag{31}$$

for the MP is derived as given in Codato and Fischetti [2006]. It ensures that the maximal expected completion time of the MP must be at least greater than or equal to c_i^{sub} if a partial assignment $\hat{J}_i$ is chosen. Because this subproblem information is in principle the same for all machines, the cut can be added for every machine.

After every subproblem has been evaluated, the maximal expected completion time of the full assignment is obtained. If this assignment is better than the incumbent solution, it becomes the incumbent solution and the MP is prevented from the generation of solutions with higher approximated objective function value. Once it becomes evident that one c_i^{sub} is larger than the objective function value of the incumbent solution the processing of the subproblems can be canceled. This is, because the solution value can only increase with further subproblem evaluations. The descending ordering of the machines and processing in the same order ensure that machines with larger approximated values are processed first. This accelerates the solution procedure. It continues with the generation of a new feasible machine assignment until no solution can be found and the incumbent solution is proven optimal.

In addition to the combinatorial Benders' cuts, also classical Benders' cuts (BC) can be derived if dual information from the subproblems are available. One way is to obtain this information from the relaxed subproblem as discussed in Rahmaniani et al. [2017]. By using a branch-and-price approach for the individual subproblems, this information can be obtained from the the root relaxation. This relaxation is stronger than or equal to using the normal relaxed subproblem [Desrosiers and Lübbecke, 2005]. If the dual information π_j for the jobs $j \in \hat{J}_i$ are available, the classical Benders' cut

$$c_{\max} \geq \sum_{j \in \hat{J}_i} \pi_j x_{ij} \qquad \forall \quad i \in I \tag{32}$$

can be added to the MP for every machine [Rahmaniani et al., 2017]. Different approaches to add these cuts are discussed in the implementation section 3.4.

3.3 Dantzig-Wolfe decomposition

A subproblem of the Benders' decomposition is decomposed into the set-partitioning formulation known as the RMP and a pricing problem where the specific notation is given in table 3.

Table 3: *Branch-and-price notations*

description	domain	type	meaning
$\hat{J}$		index	set of assigned jobs $j \subset J$
$\overline{\Omega}$		index	current set of columns n
A_{jn}	$(\hat{J}, \overline{\Omega})$	data	technological matrix
C_n	$(\overline{\Omega})$	data	cost of column n
P_j	$(\hat{J})$	data	processing time of job j
T^{pm}		data	duration of PM
T^{cm}		data	duration of CM
η		data	scale parameter of Weibull hazard function
β		data	shape parameter of Weibull hazard function
ρ_j	$(\hat{J})$	data	dual information from RMP for every job j
λ		data	dual information for first branching rule
u_n	$(\overline{\Omega})$	$\mathbb{R}^+$	selection of column n in the RMP
v_j	$(\hat{J})$	binary	selection of job j to new column in SP
c		$\mathbb{R}^+$	cost of column in the SP

The objective of the RMP is to minimize the total cost of the selected columns (33). These are represented by the technological matrix A_{jn}. Every job should be covered by exactly one column (34).

$$\min \quad \sum_{n \in \overline{\Omega}} C_n u_n \tag{33}$$

$$\sum_{n \in \overline{\Omega}} A_{jn} u_n = 1 \qquad \forall \quad j \in \hat{J} \tag{34}$$

$$u_n \geq 0 \qquad \forall \quad n \in \overline{\Omega} \tag{35}$$

For every solution of the RMP, the dual information ρ_j of the linear problem is used to generate new promising columns which should be included in the RMP. Therefore, the subproblem (expressions (36) to (38)) is solved to find a column that minimizes the negative reduced costs. The objective function consists of T^{pm}, the age of the machine in the status of the corresponding column and the resulting expected delays given this age. Further, this value is reduced by the dual information for every chosen job and also λ_1 and λ_2. The latter two correspond to the first used branching rule and discussed further on. Expression (37) is used to calculate the resulting cost that is embedded with the new column in the RMP. Note that this can be calculated in a post-processing phase.

$$\min \quad T^{pm} + \sum_{j \in \hat{J}} P_j v_j + \frac{T^{cm}}{\eta^\beta} \left(\sum_{j \in \hat{J}} P_j v_j \right)^\beta - \lambda_1 - \lambda_2 - \sum_{j \in \hat{J}} \rho_j v_j \tag{36}$$

$$c = T^{pm} + \sum_{j \in \hat{J}} P_j v_j + \frac{T^{cm}}{\eta^\beta} \left(\sum_{j \in \hat{J}} P_j v_j \right)^\beta \tag{37}$$

$$v_j \in \{0,1\} \quad \forall \quad j \in \hat{J} \tag{38}$$

Adding the new column to the RMP, another solution is generated until the subproblem indicates that no column with negative reduced cost can be found. This ends the column generation step and a solution to the relaxed problem is obtained. This can contain fractional values. To ensure the integrality condition of the underlying problem, a branching is necessary. After the branching on fractional values, new child nodes are generated which again need a column generation phase until convergence. Nodes of the branching tree can be pruned with the lower bound of the relaxation and the incumbent solution.
A natural way of branching is to branch on the number of used PM cycles as has been done in Wang et al. [2018]. If the sum over all decision variables u_n is fractional, there must be factional values of u_n. At the child nodes, one of the two inequalities in (39) is added. With the additional constraints, the dual information of λ_1 and λ_2 must be used in the subproblem as previously given in (36).

$$\sum_{n \in \overline{\Omega}} u_n \geq \left\lceil \sum_{n \in \overline{\Omega}} u_n \right\rceil \quad \text{or} \quad \sum_{n \in \overline{\Omega}} u_n \leq \left\lfloor \sum_{n \in \overline{\Omega}} u_n \right\rfloor \tag{39}$$

This branching rule does not ensure integrality because the sum of fractional values can also be integral. If this happens the branching scheme of Ryan and Foster [1981] is used. At a column with fractional u_n, two jobs j_1 and j_2 are chosen. In this paper, they are selected in index-ascending order of the jobs that are not included in the previous branching. The two jobs are branched in a conjunctive-disjunctive way.
At the left-hand branch, the constraint

$$\sum_{n \in \overline{\Omega}\ :\ A_{j_1 n} = A_{j_2 n} = 1} u_n \geq 1 \tag{40}$$

is added to the RMP and ensures that jobs j_1 and j_2 must both be present in future columns. To ensure that only columns are generated which contain both jobs, the constraint

$$v_{j_1} = v_{j_2} \tag{41}$$

is added to the subproblem.
On the other branch, the constraints

$$\sum_{n\in\overline{\Omega}\,:\,A_{j_1n}=A_{j_2n}=1} u_n \leq 0 \tag{42}$$

$$v_{j_1} + v_{j_2} \leq 1 \tag{43}$$

are added to the RMP and the subproblem, respectively. Here it is ensured that columns cannot contain both jobs.

3.4 Implementation

All approaches are implemented in Python using Gurobi 9.1.2 as solver. The monolithic model is denoted as 'MONO'. The Benders' decomposition using only the combinatorial cuts is called 'BENDERS'. The Benders' decomposition utilizing the branch-and-price approach for each subproblem and only using the combinatorial cuts is called 'BBP'.
All cuts are implemented using Lazy cuts which is a modern way of adding cuts to the problem on-the-fly as discussed in Fischetti et al. [2017]. The Lazy cuts are added while the problem is solved. For every MIP solution found, cuts are added with the obtained information using callbacks. Hence, only one problem is solved and not a problem for every cut added. The disadvantage of using Lazy cuts is that the solver does not use these cuts to calculate further valid inequalities for the problem.
Two ways of implementing classical Benders' cuts are considered in this work, namely the simultaneous and the sequential approach. As discussed in Rahmaniani et al. [2017], an efficient way of implementing classical Benders' cuts is a two-phase approach. Here, the problem is solved first by gathering all classical cuts and adding them afterwards to the model as regular constraints. This way, the Lazy cuts of the first run are considered as a part of the formulation. Hereafter, the solution process is restarted and only the combinatorial cuts are used in the second phase. The solver can now use the information better by implementing them as regular cuts. The drawback of this approach is that the root node of the branch-and-price problem has to be solved twice. In the simultaneous approach, classical Benders' cuts and combinatorial cuts are added at the same time at every subproblem. The BC is added here at the root relaxation and the CC at the optimal subproblem solution. The simultaneous approach is denoted as 'BBP-Sim' and 'BBP-Seq' is used for the sequential approach.
For the branch-and-price approach, no specific acceleration strategies (e.g. generation of multiple columns) as discussed in Desaulniers et al. [2002] are used in this work.

4 Computational study

4.1 Data generation and design of experiments

To evaluate the performance of the discussed approaches, two tests are considered. In the first test, all approaches are tested on small-sized instances. Here, the superior performance of the decomposition approaches compared to the monolithic approach becomes visible. Therefore, 'MONO' is neglected for the second test which is run for the decomposition approaches on large-sized instances.

For both tests, instances with number of machines of $|I| \in \{2, 3, 4\}$ are created. Adapted from the example of Cassady and Kutanoglu [2005], the maintenance parameters are chosen as $T^{pm} = 5$, $T^{cm} = 15$ and $\eta = 100$. Further, $\beta = 2$ as mentioned above. For the first test, number of jobs of $|J| \in \{8, 10, \ldots, 16\}$ are used and 18, 20, and 22 respectively for the second test. The processing times are sampled from a discrete uniform distribution with $U[1, \lfloor a^* \rfloor]$ to ensure that only jobs are generated that fit the machine's condition. For every combination of jobs and machines, 25 random instances are generated. Therefore, a total of 675 instances is used. The time limit for each approach in both tests is set to 3600 seconds. The capability of solving an instance in the given time limit is measured by 'term'. It is 1, if the instance can be solved to proven optimality in time and 0, otherwise. The total solution time 't-tot' and the relative time ('rt-sp') spent in the subproblem of the Benders' decomposition are reported as well. The times of the branch-and-price algorithm are not further granulated for purpose of clarity. The number of Benders' iterations are given as 'ben_iter'. The number of cuts added to the problem are given for the combinatorial cuts ('cc_count') and the classical Benders' cuts ('bc_count'). Further, a 'wins' measurement is given. Here, an approach receives a point if it can solve the instance fastest. If the solution times of the approaches are similar in a 1e-2 tolerance, all these approaches receive a point.

4.2 Results

All tests are run on an Intel(R) Core(TM) i5-6500 CPU machine at 3.2 GHz with 8 GB of RAM. The results given in tables 4 to 7 show the average performance over the 25 instances for the given configurations. The different machine configurations are aggregated in table 4 for the purpose of clarity. The instances and results can be found in the supplementary material.
For the test on small job numbers, it can be seen that all approaches are able to solve instances up to 16 jobs. Looking at the termination measurement, it becomes visible that all decomposition approaches outperform the monolithic approach. For this, the capability of solving the instance in the given time limit drops from 100% to nearly 50%. This can also be seen from 'wins' where the monolithic approach can only solve a few instances with $|J| = 16$ faster than the other approaches. These three instances with 2 machines are won clearly. However, for the other instances, the decomposition approaches clearly outperform the monolithic approach. All are capable of solving the instances using a few seconds on the smaller job numbers to nearly 100 seconds for the larger ones. Comparing the average total runtime, the sequential BBP outperforms the other approaches. The same can be seen from 'wins' were nearly 60 to 95% of the instances were solved fastest by this approach. The second approach at this measure is 'BENDERS'. It is interesting to see that this approach can win more instances than 'BBP' and 'BBP-Sim' but has a higher average runtime at $|J| = 8$ to $|J| = 14$. This reflects the greater runtime deviation of this approach.
The Benders' iterations stay nearly the same for all approaches except for 'BBP-Seq'. Here, the iterations are nearly twice as many comparing to the others. However, it can be seen that 'cc_count' is reduced through the BC. At 16 jobs, the approach is able to reduce the CC by nearly 50% with the same use of BC. The same can be seen for the other job sizes on smaller scales. The relative time spent in the subproblem is around 90% for all decomposition approaches except 'BBP-Seq' where it is around 80%. This can be related to the restart of the master problem after adding all gathered BCs which becomes more difficult to solve.
The larger instances become harder to solve with increasing number of jobs. For all machine configurations, the BBP approaches can solve the instances with 18 and 20 jobs (except

Table 4: *Results for the competitive test with small-sized instances*

$\|J\|$	sol	term	rt-sp	t-tot	wins	ben_iter	cc_count	bc_count
	MONO	1	-	19.99	0	-	-	-
	BENDERS	1	0.86	0.74	0.32	7.17	49.47	-
8	BBP	1	0.88	0.68	0.05	7.16	49.41	-
	BBP-Sim	1	0.89	0.65	0.05	6.92	49.27	41.27
	BBP-Seq	1	0.78	0.41	0.65	13.16	49.88	41.61
	MONO	0.97	-	205.27	0	-	-	-
	BENDERS	1	0.93	1.72	0.33	9.97	63.77	-
10	BBP	1	0.96	1.47	0.05	9.97	63.56	-
	BBP-Sim	1	0.95	1.56	0.03	10.44	69.79	64.48
	BBP-Seq	1	0.87	0.77	0.60	18.17	60.91	63.71
	MONO	0.91	-	525.20	0	-	-	-
	BENDERS	1	0.96	3.76	0.17	18.92	98.71	-
12	BBP	1	0.97	3.35	0.11	19.11	99.84	-
	BBP-Sim	1	0.96	3.48	0.07	18.97	99.45	96.12
	BBP-Seq	1	0.89	1.60	0.68	31.99	86.31	98.01
	MONO	0.76	-	1266.29	0	-	-	-
	BENDERS	1	0.94	11.65	0.19	48.39	188.95	-
14	BBP	1	0.94	11.24	0.01	48.25	188.45	-
	BBP-Sim	1	0.94	11.57	0.01	48.95	194.23	191.41
	BBP-Seq	1	0.83	5.20	0.79	73.33	129.31	199.93
	MONO	0.53	-	2012.65	0.04	-	-	-
	BENDERS	1	0.88	79.07	0.01	163.72	521.19	-
16	BBP	1	0.88	82.22	0	164.35	525.01	-
	BBP-Sim	1	0.88	81.35	0	162.40	519.40	516.87
	BBP-Seq	1	0.70	20.18	0.95	227.31	274.43	515.92

'BBP-Seq' at 2 machines and 20 jobs). The normal Benders' approach struggles here by only solving nearly 45% of the instances for 2 machines. For 22 jobs, the termination of all approaches is under 10% for 2 and 3 machines where for 4 machines the capability of solving these instances increases for all BBP approaches.
The low 'term' measurement is also reflected by an increase of average total runtime which is close to the given time limit for the larger instances. The superior performance of the BBP approaches in comparison to 'BENDERS' can also be seen here. The average runtime is several times larger for 18 and 20 jobs. In contrast to this, 'BENDERS' shows a good 'wins' performance. On average, it can win 40% of the instances for 18 and 20 jobs. This means that 'BENDERS' can solve some instances faster than the BBP approaches but needs much more effort on others to solve them at all. The runtime also shows that the performance of 'BBP-Seq' does not scale well with an increasing job number. The approach is nearly two to four times faster on small instances and up to two times (4 machines and 20 jobs) slower than the other BBP approaches. This can be due to the repeated solving of the subproblem root node as discussed above. However, for 2 machines the runtime performance is comparable. Here, the 'wins' measurements indicate good performance which cannot be seen at other machine configurations. Comparing the CC-BC ratio, the same as for the small instances

Table 5: *Results for the competitive test with large-sized instances with 2 machines*

$\|J\|$	sol	term	rt-sp	t-tot	wins	ben_iter	cc_count	bc_count
18	BENDERS	1	0.99	265.91	0.44	1072.56	2434.72	-
	BBP	1	0.98	173.33	0.12	1072.48	2430.40	-
	BBP-Sim	1	0.97	173.92	0	1072.68	2430.56	2430.48
	BBP-Seq	1	0.93	179.86	0.44	1378.96	706.32	2445.04
20	BENDERS	0.44	0.98	2882.90	0.24	3278.20	7166.56	-
	BBP	1	0.95	1468.35	0.28	4274.28	9394.64	-
	BBP-Sim	1	0.93	1509.50	0.08	4274.04	9368.00	9367.60
	BBP-Seq	0.96	0.88	1499.31	0.40	5438.40	2514.96	9608.56
22	BENDERS	0	1	3640.17	0	2041.04	4701.36	-
	BBP	0.08	0.94	3565.02	0.08	7374.32	17026.48	-
	BBP-Sim	0.08	0.91	3595.86	0	7072.24	16275.76	16275.60
	BBP-Seq	0.04	0.82	3569.60	0	10688.00	80.80	24246.00

Table 6: *Results for the competitive test with large-sized instances with 3 machines*

$\|J\|$	sol	term	rt-sp	t-tot	wins	ben_iter	cc_count	bc_count
18	BENDERS	0.96	0.76	242.39	0.32	463.96	1644.36	-
	BBP	1	0.74	48.64	0.24	461.88	1639.56	-
	BBP-Sim	1	0.73	45.99	0.40	407.28	1462.08	1459.80
	BBP-Seq	1	0.59	66.85	0.04	580.04	764.40	1364.16
20	BENDERS	0.8	0.58	1070.52	0.64	1483.96	5078.16	-
	BBP	1	0.56	620.66	0.16	1920.12	6447.12	-
	BBP-Sim	1	0.48	734.83	0.16	1778.44	5953.08	5951.88
	BBP-Seq	1	0.43	939.85	0.04	2631.24	3405.84	5555.16
22	BENDERS	0.08	0.46	3527.68	0.08	5101.80	17087.28	-
	BBP	0.04	0.53	3585.51	0	4894.84	17109.84	-
	BBP-Sim	0	0.49	3601.15	0	4532.08	15831.24	15830.28
	BBP-Seq	0	0.31	3605.07	0	5473.44	786.00	18368.64

becomes visible. The use of BC in the sequential setting reduces the use of CC. Note here that for larger instances where the approach does not always terminate, a low 'cc_count' can be explained by a termination in the first phase.
Comparing 'BBP' with 'BBP-Sim', it seems that 'BBP' performs better on smaller $|I|$ and 'BBP-Sim' for larger machine number. A small reduction in Benders' iterations and the number of the combinatorial cuts can be seen due to the utilization of BC. On the opposite, 'BBP-Sim' has less relative subproblem time. This shows that more effort is needed in the master problem due to the additional cuts.
Overall, the proposed decomposition approaches outperform the monolithic model even on small instances. The 'BENDERS' approach can be faster than the other approaches on some instances. On others, it shows bad capability of solving them. 'BBP' and 'BBP-Sim' show good average performance with a slight advantage on 2 machines for the 'BBP' and for the 'BBP-Sim' on four machines. To evaluate a clearer performance impact of using the simultaneous addition of BC, may be the focus of ongoing work. 'BBP-Seq' performs well

Table 7: *Results for the competitive test with large-sized instances with 4 machines*

$\|J\|$	sol	term	rt-sp	t-tot	wins	ben_iter	cc_count	bc_count
18	BENDERS	1	0.56	180.49	0.32	134.32	778.40	-
	BBP	1	0.41	19.57	0.28	131.80	764.80	-
	BBP-Sim	1	0.37	21.01	0.40	128.32	736.96	730.08
	BBP-Seq	1	0.29	38.18	0	193.00	494.98	730.40
20	BENDERS	0.72	0.51	1341.25	0.44	348.28	1727.04	-
	BBP	1	0.26	233.29	0.20	517.36	2481.12	-
	BBP-Sim	1	0.23	292.77	0.28	447.00	2244.48	2239.04
	BBP-Seq	1	0.20	480.33	0.08	596.44	1256.64	1995.68
22	BENDERS	0.04	0.66	3597.59	0	980.04	4222.08	-
	BBP	0.4	0.12	3263.97	0.20	1993.84	9028.32	-
	BBP-Sim	0.36	0.13	2738.76	0.32	1716.60	7680.16	7672.64
	BBP-Seq	0.28	0.08	3107.29	0	1952.00	2418.72	6788.16

on small instances but scales worse on larger instances compared to 'BBP' and 'BBP-Seq'. The impact of BC is interesting to see at this approach but a different way to cope with the repeated solving of the root node is needed here.

5 Conclusion

The IPSMP is an integrated problem combining the scheduling of production jobs and the planning of maintenance activities on machines. Both compete for machine time during the planning horizon while one depletes the machine condition and the other restores it. The maintenance activities are necessary because a machine is more likely to fail by increasing operational time. These failures cause random delays which can be expressed as a combination of the expected number of failures and the duration for a corrective maintenance.
As the deterministic version of this problem is discussed well for the parallel-machine environment, the stochastic case lacks this consideration. Further, less work is done for the introduction of mathematical programming approaches to the stochastic case in general. To close this gap, this work proposes a monolithic mathematical model of the problem for an identical parallel-machine environment. The model is non-linear in nature and contains difficult quadratic constraints.
To accelerate the solution process, different ways to decompose the problem are discussed. The used Benders' decomposition transforms the non-linear constraints to a well-manageable quadratic objective function in the subproblems. This can be decomposed further by using a Dantzig-Wolfe decomposition leaving a strong linear relaxation in the Benders' subproblem. Hereby, the quadratic objective function is shifted to the subproblem of the used branch-and-price algorithm. With the root relaxation of this approach, classical Benders' cuts are used in the master problem. Also different ways to implement these cuts are discussed in this paper.
The results show a superior performance of the decomposition approaches compared to the monolithic model on small-sized instances. For larger ones, the 'BBP' and 'BBP-Sim' show the best average performance while leaving the open question of the impact of classical Benders cut on these two approaches. Further, the improvement of the sequential approach

seems to be a promising field for future research.

References

Abdelrahim, E. H. and Vizvári, B. [2017]. Simultaneous Scheduling of Production and Preventive Maintenance on a Single Machine, *Arabian Journal for Science and Engineering* **42**(7): 2867–2883.

Bajestani, M. A., Banjevic, D. and Beck, J. C. [2014]. Integrated maintenance planning and production scheduling with Markovian deteriorating machine conditions, *International Journal of Production Research* **52**(24): 7377–7400.

Benders, J. F. [1962]. Partitioning procedures for solving mixed-variables programming problems, *Numerische Mathematik* **4**(1): 238–252.

Cassady, C. R. and Kutanoglu, E. [2003]. Minimizing Job Tardiness Using Integrated Preventive Maintenance Planning and Production Scheduling, *IIE Transactions* **35**(6): 503–513.

Cassady, C. R. and Kutanoglu, E. [2005]. Integrating Preventive Maintenance Planning and Production Scheduling for a Single Machine, *IEEE Transactions on Reliability* **54**(2): 304–309.

Chen, J.-S. [2006a]. Optimization models for the machine scheduling problem with a single flexible maintenance activity, *Engineering Optimization* **38**(1): 53–71.

Chen, J.-S. [2006b]. Using integer programming to solve the machine scheduling problem with a flexible maintenance activity, *Journal of Statistics and Management Systems* **9**(1): 87–104.

Codato, G. and Fischetti, M. [2006]. Combinatorial Benders' Cuts for Mixed-Integer Linear Programming, *Operations Research* **54**(4): 756–766.

Cui, W. [2020]. Approximate Approach to Deal with the Uncertainty in Integrated Production Scheduling and Maintenance Planning, *Journal of Shanghai Jiaotong University (Science)* **25**(1): 106–117.

Cui, W., Lu, Z., Li, C. and Han, X. [2018]. A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops, *Computers & Industrial Engineering* **115**: 342–353.

Desaulniers, G., Desrosiers, J. and Solomon, M. M. [2002]. Accelerating Strategies in Column Generation Methods for Vehicle Routing and Crew Scheduling Problems, *in* R. Sharda, S. Voß, C. C. Ribeiro and P. Hansen (eds), *Essays and Surveys in Metaheuristics*, Vol. 15 of *Operations Research/Computer Science Interfaces Series*, Springer US, Boston, MA, pp. 309–324.

Desrosiers, J. and Lübbecke, M. E. [2005]. A Primer in Column Generation, *in* G. Desaulniers, J. Desrosiers and M. M. Solomon (eds), *Column Generation*, Springer-Verlag, New York, pp. 1–32.

Fischetti, M., Ljubić, I. and Sinnl, M. [2017]. Redesigning Benders Decomposition for Large-Scale Facility Location, *Management Science* **63**(7): 2146–2162.

Hadidi, L. A., Turki, U. M. A. and Rahim, A. [2012a]. Integrated models in production planning and scheduling, maintenance and quality: a review, *International Journal of Industrial and Systems Engineering* **10**(1): 21.

Hadidi, L. A., Turki, U. M. A. and Rahim, M. A. [2012b]. Joint job scheduling and preventive maintenance on a single machine, *International Journal of Operational Research* **13**(2): 174.

Huo, Y. and Zhao, H. [2011]. Bicriteria scheduling concerned with makespan and total completion time subject to machine availability constraints, *Theoretical Computer Science* **412**(12-14): 1081–1091.

Kaabi, J. and Harrath, Y. [2014]. A Survey of Parallel Machine Scheduling under Availability Constraints, *International Journal of Computer and Information Technology* pp. 238–245.

Laporte, G. and Louveaux, F. V. [1993]. The integer L-shaped method for stochastic integer programs with complete recourse, *Operations Research Letters* **13**(3): 133–142.

Lee, C.-Y. and Chen, Z.-L. [2000]. Scheduling jobs and maintenance activities on parallel machines, *Naval Research Logistics* **47**(2): 145–165.

Li, G., Liu, M., Sethi, S. P. and Xu, D. [2017]. Parallel-machine scheduling with machine-dependent maintenance periodic recycles, *International Journal of Production Economics* **186**: 1–7.

Liao, W., Chen, M. and Yang, X. [2017]. Joint optimization of preventive maintenance and production scheduling for parallel machines system, *Journal of Intelligent & Fuzzy Systems* **32**(1): 913–923.

Ma, Y., Chu, C. and Zuo, C. [2010]. A survey of scheduling with deterministic machine availability constraints, *Computers & Industrial Engineering* **58**(2): 199–211.

Mokhtari, H., Mozdgir, A. and Kamal Abadi, I. N. [2012]. A reliability/availability approach to joint production and maintenance scheduling with multiple preventive maintenance services, *International Journal of Production Research* **50**(20): 5906–5925.

Rahmaniani, R., Crainic, T. G., Gendreau, M. and Rei, W. [2017]. The Benders decomposition algorithm: A literature review, *European Journal of Operational Research* **259**(3): 801–817.

Rebai, M., Kacem, I. and Adjallah, K. H. [2013]. Scheduling jobs and maintenance activities on parallel machines, *Oper Res Int J* **13**(3): 363–383.

Ryan, D. and Foster, B. [1981]. An Integer Programming Approach to Scheduling, *Computer Scheduling of Public Transport* (1): 269–280.

Sadiqi, A., El Abbassi, I., El Barkany, A. and El Biyaali, A. [2018]. Joint Scheduling of Jobs and Variable Maintenance Activities in the Flowshop Sequencing Problems: Review, Classification and Opportunities, *International Journal of Engineering Research in Africa* **39**: 170–190.

Schmidt, G. [2000]. Scheduling with limited machine availability, *European Journal of Operational Research* **121**(1): 1–15.

Seif, J., Dehghanimohammadabadi, M. and Yu, A. J. [2019]. Integrated preventive maintenance and flow shop scheduling under uncertainty, *Flexible Services and Manufacturing Journal* **153**(3): 534.

Shen, J. and Zhu, Y. [2019]. A parallel-machine scheduling problem with periodic maintenance under uncertainty, *Journal of Ambient Intelligence and Humanized Computing* **10**(8): 3171–3179.

Sortrakul, N. and Cassady, C. R. [2007]. Genetic algorithms for total weighted expected tardiness integrated preventive maintenance planning and production scheduling for a single machine, *Journal of Quality in Maintenance Engineering* **13**(1): 49–61.

Sun, K. and Li, H. [2010]. Scheduling problems with multiple maintenance activities and non-preemptive jobs on two identical parallel machines, *International Journal of Production Economics* **124**(1): 151–158.

von Hoyningen-Huene, W. and Kiesmüller, G. P. [2015]. Evaluation of the expected makespan of a set of non-resumable jobs on parallel machines with stochastic failures, *European Journal of Operational Research* **240**(2): 439–446.

Wang, L.-Y., Huang, X., Ji, P. and Feng, E.-M. [2014]. Unrelated parallel-machine scheduling with deteriorating maintenance activities to minimize the total completion time, *Optimization Letters* **8**(1): 129–134.

Wang, S. [2013]. Bi-objective optimisation for integrated scheduling of single machine with setup times and preventive maintenance planning, *International Journal of Production Research* **51**(12): 3719–3733.

Wang, S. and Liu, M. [2013]. A branch and bound algorithm for single-machine production scheduling integrated with preventive maintenance planning, *International Journal of Production Research* **51**(3): 847–868.

Wang, S. and Liu, M. [2016]. Two-machine flow shop scheduling integrated with preventive maintenance planning, *International Journal of Systems Science* **47**(3): 672–690.

Wang, T., Baldacci, R., Lim, A. and Hu, Q. [2018]. A branch-and-price algorithm for scheduling of deteriorating jobs and flexible periodic maintenance on a single machine, *European Journal of Operational Research* **271**(3): 826–838.

Xu, D. and Yang, D.-L. [2013]. Makespan minimization for two parallel machines scheduling with a periodic availability constraint: Mathematical programming model, average-case analysis, and anomalies, *Applied Mathematical Modelling* **37**(14-15): 7561–7567.

Yang, S.-J. [2013]. Unrelated parallel-machine scheduling with deterioration effects and deteriorating multi-maintenance activities for minimizing the total completion time, *Applied Mathematical Modelling* **37**(5): 2995–3005.

Yulan, J., Zuhua, J. and Wenrui, H. [2008]. Multi-objective integrated optimization research on preventive maintenance planning and production scheduling for a single machine, *The International Journal of Advanced Manufacturing Technology* **39**(9-10): 954–964.

Chapter 5

Article 4: Decomposition approach for integrated production scheduling and maintenance planning of a cyclic flowshop with random failures

Decomposition approach for integrated production scheduling and maintenance planning of a cyclic flowshop with random failures

Sven Pries

Abstract

In this paper, an approach for the integrated production-scheduling and maintenance-planning problem in a multi-machine environment with random failures is presented. The combination of job scheduling with the insertion of preventive maintenance activities is used to reduce the drawbacks of random failures in an industrial production setting and to save costs. A cyclic flowshop scheduling problem with blocking is considered. The objective is to minimize the expected maximal cycle time which hedges better against the impact of random failures. To solve this stochastic programming problem, a decomposition approach combined with a simulative approximation is presented. It uses combinatorial Benders' cuts and problem-specific properties in the solution procedure. The performance of the used cuts which enable the utilization of lower bounds is evaluated through a computational study and shows a strong impact on runtime.

Keywords: scheduling, maintenance, flowshop, random failures, decomposition

1 Introduction

In modern industrial plants, work is carried out day and night. To ensure the high availability of production capacities, maintenance activities are needed. They prevent the production from random failures which cause unexpected delays or even line stoppages in flow production systems. If these activities need to be carried out during working shifts, maintenance activities and production jobs compete for machine time. The first restores the machine's condition whereby the latter depletes it. The joint planning of both tasks yields the best potential for performance improvements in an uncertain environment like an industrial production. This is because the inter-dependencies of both decisions can be considered.
In many production scheduling papers, it is however assumed that machines are available throughout production [Ma et al., 2010] and no maintenance activities are necessary. A way to consider these activities is to introduce unavailable intervals or periods into the schedule where no job can be scheduled.
The literature on scheduling with limited machine availability can be classified into the deterministic case and the stochastic case. In the first case, the times of the unavailable intervals are given beforehand through a previous sequential planning step. They can be either fixed or in a given time window. For the second case, maintenance periods are scheduled flexibly in number and time to reduce the risk of unplanned random failures. The latter are due to the degradation of machines caused by job processing. Both cases are discussed in the reviews of Hadidi et al. [2012a] and Sadiqi et al. [2018].

Deterministic scheduling with fixed availability constraints has received most of the attention in prior research since the 1990s. Start and end times of non-available intervals can be viewed as fixed and known in advance. Therefore, it is possible to model planned maintenance of machines throughout the planning horizon. The jobs are scheduled then in a way to prevent interference with these intervals. For this group of problems, comprehensive surveys are provided by Sanlaville and Schmidt [1998], Schmidt [2000] and Ma et al. [2010].
For the flowshop case, Lee [1997] studies a two-machine system, minimizing the maximal completion time with resumable jobs and extends this work in Lee [1999] for the semi-resumable case. Here, a resumable job means that a job interrupted by the non-available interval can be continued afterwards without penalty (e.g., additional processing time) or in the semi-resumable case with a penalty. Cheng and Wang [2000] provide an improved heuristic for the two-machine flowshop with an unavailability period on the first machine for the heuristic given by Lee [1997]. Kubiak et al. [2002] solve the two-machine flowshop problem with a branch-and-bound algorithm. Ng and Kovalyov [2004] study a problem in which the interval can be on the first or on the second machine and provide a fully polynomial-time approximation scheme. Breit [2004] studies a two-machine flowshop minimizing the makespan where the unavailability period is on the second machine. Further, he provides a polynomial-time approximation scheme for the same problem with an unavailability period on the first machine in Breit [2006]. Aggoune [2004] solves the multi-machine flowshop problem with availability constraints using a genetic algorithm and a tabu search. Aggoune and Portmann [2006] extend a geometric approach developed for the two-machine case to solve the m-machine problem. The problem with non-resumable jobs and an unavailability interval on the first machine is studied by Hadda et al. [2010]. Also Hadda [2012] provides a polynomial-time approximation scheme under the resumable scenario where several unavailability periods are on the first machine. Shoaardebili and Fattahi [2015] study a three-stage assembly flowshop scheduling problem minimizing total weighted completion time and sum of weighted tardiness and earliness simultaneously. They used a non-dominated sort genetic algorithm and a multi-objective simulated annealing to find Pareto-optimal solutions. Hnaien et al. [2015] propose a branch-and-bound algorithm to the two-machine flowshop problem with an availability constraint on the first machine and minimization of the makespan.
In other deterministic cases of scheduling with availability constraints, the times are not fixed and can be scheduled in a flexible manner. Aggoune [2004] solves two variants of this scheduling problem where, for one variant, the unavailability period has to be scheduled in a given time window. For the other variant, times are fixed. Kubzin and Strusevich [2005] use floating maintenance in a two-machine flowshop environment minimizing the makespan. The duration of this maintenance interval is given by a non-decreasing function of the interval's starting time. Kubzin and Strusevich [2006] study a two-machine system minimizing the total completion time where each machine has to be maintained once in the planning horizon. Allaoui et al. [2008] solve the two-machine case with a non-available period on either of them. Vahedi-Nouri et al. [2014] solve a general flowshop problem with position-based learning effects and multiple availability constraints. They assume that after the processing of a given number of production jobs, a maintenance activity must be scheduled.
When random failures resulting from the depletion of machines (e.g. abrasion or staining) are considered, a machine with long total operational time (meaning a machine of older age) is more likely to fail. Preventive maintenance (PM) activities can be scheduled in the planning horizon to reduce the age and therefore the risk of unplanned unavailability periods caused by failures. During these periods, the machine undergoes a corrective maintenance (CM). Cassady and Kutanoglu [2003] first consider the simultaneous planning of production and

maintenance activities with the idea of minimal repair at random failures. Here, a CM does not change the machine's age. They study a single machine minimizing the total weighted expected tardiness where the machine can fail only once when processing a job. The only way to set back the machine's age to an as-good-as-new state is to schedule a PM. Cassady and Kutanoglu [2005] extend the model for minimizing the total weighted expected completion time and considering multiple failures. They solve the model using full enumeration for small instances. Sortrakul and Cassady [2007] present a genetic algorithm for the problem of Cassady and Kutanoglu [2003]. Yulan et al. [2008] extend the single-machine case by five objectives and solve the multi-objective problem using a genetic algorithm. Hadidi et al. [2012b] solve the problem with total weighted expected completion time using a non-linear binary formulation. Wang [2013] studies the problem for a bi-objective optimization considering total expected completion time and expected number of failures, and proposes a genetic algorithm as solution approach. Wang and Liu [2013] suggest a branch-and-bound algorithm for a single machine with multiple failures and minimizing the total weighted expected completion time. Cui [2020] studies robustness of quality and solution by introducing idle times for the single-machine problem and solves it with a three-stage algorithm, involving a stage with a gradient-descent algorithm based on a surrogate measure.
Less work has been done for the stochastic problem in the flowshop environment. Wang and Liu [2016] present a genetic algorithm to solve large-scale instances of the two-machine flowshop problem minimizing the maximal expected completion time. They use the expected-value problem instead of a stochastic programming approach. Khatami and Zegordi [2017] solve the bi-objective problem with minimization of makespan and unavailability using ant colony optimization. Cui et al. [2018] study the robustness in a multi-machine flowshop, not considering the multi-failure case. They propose that it can be considered by the use of a simulation approach but opting for a surrogate measure to approximate the value for the one-failure case. A two-loop algorithm is used to solve the bi-objective problem. Seif et al. [2019] propose a two-stage stochastic mixed-integer program for the problem minimizing the total expected costs. For solving large instances, a simulation-optimization approach is used. By reviewing the literature, it becomes evident that much work has been done for the single-machine environment which can be used as a building block for more complex problems. Less work has been done for the flowshop environment. Also simplifications are conducted to the stochastic nature of the problem when considering a flowshop. Either the use of the expected-value problem [Wang and Liu, 2016] or the consideration of only one failure [Cui et al., 2018] at producing a job is studied in recent works. The solution approaches range from exact full enumeration approaches to meta-heuristics approaches. Less work is done by using mixed-integer programming (MIP) to this problem. The only approach that can be mentioned here is the work of Seif et al. [2019].
The problem presented in this work considers a stochastic programming approach with simulation for the flowshop problem by minimizing the expected maximal cycle time. This objective is different to the ones in the previous works in two ways. First, a small optimal sequences can be repeated periodically to express a large cyclic schedule if the maximal cycle time instead of the makespan is considered. The makespan represent the maximal completion time of all jobs. The maximal cycle time for a cyclic schedule is the maximum of the production intervals of the individual machines ranging from the start of the first job to the completion of the last. With this objective, only a small problem must be solved to improve the performance of large schedules. Second, the uncertainty consideration with a stochastic programming approach is used. This hedges better, compared to the expected-value problem, against the impact of random failures (as further detailed in section 3).

This is because changes of the critical machine due to failures are considered. The multi-failure case is modeled which reflects the real performance more accurately. However, the possible scenarios and therefore the computational effort evaluating all of them increases dramatically. By assuming blocking, the environmental inter-dependencies are increased because no buffers between the production stages are considered. Therefore, the impact of random failures is amplified. With blocking, a machine has to wait for the successor machine to finish before starting the next job. Hence, a failure not only affecting the successors but also the predecessors. To contribute to the range of used solution approaches for this problem, a heuristic decomposing the problem in a Benders' fashion (see Rahmaniani et al. [2017]) is proposed. Tailored lower bounds are presented to improve the solution performance. Combinatorial Benders' cuts are used to exchange information as well as a Monte-Carlo simulation to evaluate the expected maximal cycle time.
The remainder of this work is structured as follows. First, in section 2, the problem under consideration is introduced. In section 3, problem-specific properties are presented that are incorporated as lower bounds in the solution approach (section 4). In the computational study in section 5, the performance of the lower bounds compared with a full enumerative benchmark approach is shown. A conclusion of the paper and an outlook on further research is provided in section 6.

2 Problem description

The problem discussed throughout this paper can be classified as an integrated production-scheduling and maintenance-planning problem (IPSMP) for a cyclic permutation flowshop with random failures and blocking. The objective is to minimize the expected maximal cycle time over all machines.
In the Gantt chart in figure 1, an example instance of the underlying problem is shown.

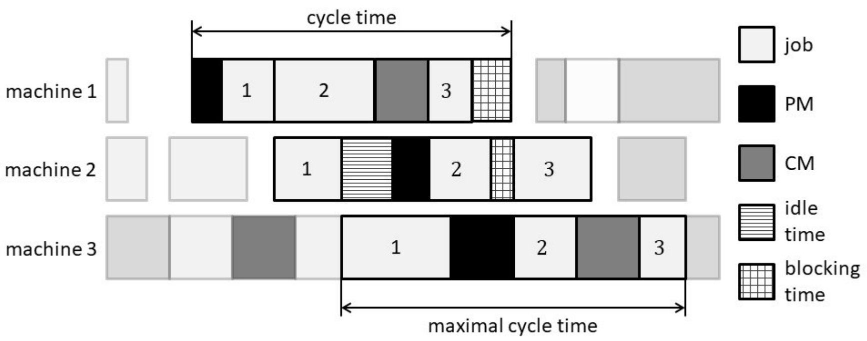

Figure 1: *Problem example instance*

A cyclic production plan is a partial plan for a given set of jobs (here 1 to 3) which can be repeated to yield longer production plans. The same partial plan can be seen hinted on the left and the right of the example and it must be ensured that these plans fit together. Machines can be idle while waiting for jobs to process (machine 2 after job 1) or can be blocked to pass the workpiece because the successor machine is still busy (machine 1 after job 3 and machine 2 after job 2). The cycle time of an individual machine ranges from the start of production to the point in time when the last job is passed. The maximal cycle time determines the duration of the partial plan and the responsible machine is called the critical machine (here machine 3). Throughout the production, the machines can fail and must be repaired. This delays the job completion. The example provides only one possible failure

scenario with failures at the first and the third machine at the production of the second job. An older machine, which means a machine which has been in use longer, is more likely to fail. Hence, PMs (dark rectangles) can be inserted prior to every job to reduce the age and therefore the probability of failures. However, these activities block the machine for job processing. To ensure the fit of the cyclic plans, the age at the beginning and the end of a cycle must be the same for every machine.
In general, there exists a set of jobs J which has to be scheduled on a set of machines I. A permutation flowshop is assumed. Hence, every job has to visit the machines in the same and predefined sequence. The jobs are assigned to a set of ranks K where the assignment of jobs to ranks is the same for all machines. With the uncertainty of random failures during job processing, there exists a set of possible failure scenarios W. A scenario w is defined by a possible combination of the realized number of failures ξ_{ik}^w at all ranks on all machines.
To assign jobs to ranks, the binary decision variable

$$x_{jk} = \begin{cases} 1, & \text{if job } j \text{ is assigned to rank } k \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

is used. Each job has a processing time P_{ij} which indicates the length of an operation of job j on machine i. All processing times are assumed to be integers. The operations on all machines should not overlap with each other. Hence, there exists a starting time s_{ik}^w and a completion time c_{ik}^w for every rank k on machine i in scenario w. Since the objective is to minimize the expected maximal cycle time over all machines, the variable $ct_{\max}^w$ describes this maximal cycle time given a scenario w.
The probability of a scenario w is the product of the individual failure probabilities Pr_{ik}^w. An age-dependent policy [Wang, 2002] is assumed. Pr_{ik}^w is a function that determines the probability of a realized number of failures given the machine's age during the production of a job. To model the age prior and past to the production, two variables as_{ik} and ac_{ik} are introduced to represent the age before and after the production of the job on rank k at machine i. The machine is minimally repaired at failure. This means that the age does not change if a failure occurs and the completion time of the job is delayed by a duration of T_i^{cm}. With the property of minimal repair, the time between start and completion of a job with a given age before and after processing can be seen as a non-homogeneous Poisson process with a Weibull hazard function

$$z(t) = \frac{\beta_i}{\eta_i^{\beta_i}} t^{\beta_i - 1} \tag{2}$$

as given in Cassady and Kutanoglu [2005] with the scale parameter $\eta_i > 0$ and shape parameter $\beta_i > 1$. The latter is used to model the increasing probability for failures on individual machine i throughout its operational time. This function is common in maintenance-planning problems and models with the occurrence of failures with increasing rates [Cassady and Kutanoglu, 2003]. The probabilities $Pr_{ik}^w(\xi_{ik}^w, as_{ik}, ac_{ik})$ given the number of realized failures ξ_{ik}^w, as_{ik} and ac_{ik} can be expressed with the Poisson distribution function (poi) as

$$Pr_{ik}^w(\xi_{ik}^w, as_{ik}, ac_{ik}) = poi(\xi_{ik}^w, m(as_{ik}, ac_{ik})) = \frac{m(as_{ik}, ac_{ik})^{\xi_{ik}^w} e^{-m(as_{ik}, ac_{ik})}}{\xi_{ik}^w!} \tag{3}$$

where $m(as_{ik}, ac_{ik})$ is the expected number of failures [Cassady and Kutanoglu, 2005] during production given by the hazard function as

$$m(as_{ik}, ac_{ik}) = \left(\frac{ac_{ik}}{\eta_i}\right)^{\beta_i} - \left(\frac{as_{ik}}{\eta_i}\right)^{\beta_i} = \int_{as_{ik}}^{ac_{ik}} z(t)\, dt. \tag{4}$$

To reduce the increasing age, a PM is needed to bring back the machine to an as-good-as-new state. Here, the age is set back to zero. The duration of a PM is T_i^{pm}. Because the fit of the starting and end ages of the schedule is assumed, at least one PM is needed on each machine. It is assumed that if a PM is scheduled at rank k on machine i it is always scheduled prior to the job at this rank. The binary decision variable y_{ik} indicates whether a PM is scheduled ($y_{ik} = 1$) or not ($y_{ik} = 0$). Pr_{ik}^w depends on as_{ik} and ac_{ik} which can be controlled through the decisions on PM with y_{ik}. Hence, the uncertainty in this problem is decision-dependent. The variables ps_{ik}^w and pc_{ik}^w define the starting and completion time of the maintenance activity and are present at every rank k at machine i in scenario w. If a PM is scheduled, $pc_{ik}^w = ps_{ik}^w + T^{pm}$ and $pc_{ik}^w = ps_{ik}^w$ otherwise. A non-resumable case is considered here where a PM cannot interrupt the processing of a job. Hence, a PM must not overlap with the job processing intervals.

Table 1: *Notations*

description	domain	type	meaning
I		index	set of machines i
J		index	set of jobs j
K		index	set of ranks k
W		index	set of scenarios w
P_{ij}	(I, J)	data	processing time of job j on machine i
T_i^{pm}	(I)	data	time duration of PM activity on machine i
T_i^{cm}	(I)	data	time duration of CM activity on machine i
η_i	(I)	data	scale parameter of Weibull hazard function on machine i
β_i	(I)	data	shape parameter of Weibull hazard function on machine i
M_i	(I)	data	BigM for machine i
ξ_{ik}^w	(I, K, W)	data	number of failures happening on rank k on machine i in scenario w
x_{jk}	(J, K)	binary	assignment of job j to rank k
s_{ik}^w	(I, K, W)	$\mathbb{R}^+$	starting time of rank k on machine i in scenario w
c_{ik}^w	(I, K, W)	$\mathbb{R}^+$	completion time of rank k on machine i in scenario w
$ct_{\max}^w$	(W)	$\mathbb{R}^+$	maximal cycle time in scenario w
y_{ik}	(I, K)	binary	PM decision for rank k on machine i
as_{ik}	(I, K)	$\mathbb{R}^+$	age before processing job on rank k on machine i
ac_{ik}	(I, K)	$\mathbb{R}^+$	age after processing job on rank k on machine i
ps_{ik}^w	(I, K, W)	$\mathbb{R}^+$	starting time of PM at rank k on machine i in scenario w
pc_{ik}^w	(I, K, W)	$\mathbb{R}^+$	completion time of PM at rank k on machine i in scenario w
Pr_{ik}^w	(I, K, W)	function	individual failure probability for rank k on machine i in scenario w

Given the combined notations in table 1, a formulation of the given problem can be expressed as the monolithic model (MONO)

$$\min \quad \sum_{w \in W} ct_{\max}^w \prod_{i \in I, k \in K} Pr_{ik}^w(\xi_{ik}^w, as_{ik}, ac_{ik}) \tag{5}$$

$$ct_{\max}^w \geq s_{i+1|K|}^w - ps_{i1}^w \qquad \forall \quad i \in I \setminus |I|, w \in W \tag{6}$$

$$ct_{\max}^w \geq c_{|I||K|}^w - ps_{|I|1}^w \qquad \forall \quad w \in W \tag{7}$$

$$\sum_{j \in J} x_{jk} = 1 \qquad \forall \quad k \in K \tag{8}$$

$$\sum_{k \in K} x_{jk} = 1 \qquad \forall \quad j \in J \tag{9}$$

$$ps^w_{ik+1} \geq c^w_{ik} \qquad \forall \quad i \in I, k \in K \setminus |K|, w \in W \tag{10}$$

$$pc^w_{ik} = ps^w_{ik} + T^{pm}_i y_{ik} \qquad \forall \quad i \in I, k \in K, w \in W \tag{11}$$

$$s^w_{ik} \geq pc^w_{ik} \qquad \forall \quad i \in I, k \in K, w \in W \tag{12}$$

$$c^w_{ik} = s^w_{ik} + T^{cm}_i \xi^w_{ik} + \sum_{j \in J} x_{jk} P_{ij} \qquad \forall \quad i \in I, k \in K, w \in W \tag{13}$$

$$s^w_{i+1k} \geq c^w_{ik} \qquad \forall \quad i \in I \setminus |I|, k \in K, w \in W \tag{14}$$

$$ps^w_{ik+1} \geq s^w_{i+1k} \qquad \forall \quad i \in I \setminus |I|, k \in K \setminus |K|, w \in W \tag{15}$$

$$ac_{ik} = as_{ik} + \sum_{j \in J} x_{jk} P_{ij} \qquad \forall \quad i \in I, k \in K \tag{16}$$

$$as_{ik+1} \geq ac_{ik} - y_{ik+1} M_i \qquad \forall \quad i \in I, k \in K \setminus |K| \tag{17}$$

$$as_{i1} \geq ac_{i|K|} - y_{i1} M_i \qquad \forall \quad i \in I \tag{18}$$

$$y_{11} = 1 \tag{19}$$

$$ct^w_{\max}, ps^w_{ik}, pc^w_{ik}, s^w_{ik}, c^w_{ik}, as_{ik}, ac_{ik} \geq 0 \qquad \forall \quad i \in I, k \in K, w \in W \tag{20}$$

$$x_{jk}, y_{ik} \in \{0, 1\} \qquad \forall \quad i \in I, j \in J, k \in K. \tag{21}$$

The objective function (5) expresses the expectation of $ct^w_{\max}$ which is the maximum of the cycle times for every machine i in scenario w. This is determined by the difference of the time when the machine returns to idle after processing the job on the last rank and the starting time of the first possible PM. For every machine, except the last one, machines become idle after the last job if the next machine starts processing the finished job (6). For the last machine, the machine becomes idle after completion of the last job (7). Equations (8) and (9) ensure that exactly one job is assigned to each rank.
With expressions (10) - (15), no job or PM interferes with another and the precedence relationship is met. A PM can only start, if the previous rank on the same machine has been completed (10). The time of PM completion is equal to the starting time if no PM is scheduled prior to the rank and the starting time plus the time for this activity otherwise (11). To start the processing of a rank, the prior PM has to be finished (12). The completion time of a rank depends on the starting time, the processing time of the job assigned to this rank and the stochastic delays. The latter result from the number of failures occurring in a scenario combined with the CM duration (13). Every successor machine rank can only start if the same rank has been completed on the previous machine (14). Inequality (15) ensures that the blocking assumption of machines is met. Hence, every PM on the predecessor

machine cannot start until the successor machine starts the production of the job newly located there.
Expressions (16)-(18) describe the age relationships. The age after a rank depends on the age prior to the production and the processing time of the assigned job (16). Inequality (17) ensures that the age before a rank can be set back to zero if a PM is scheduled. For this, a BigM formulation is needed where the parameter M_i is a sufficiently large number to allow setting the age of machine i to zero. The maximal possible age for every machine can be used here, which is the sum of all processing times on this machine. Inequality (18) guarantees that the cyclic idea is fulfilled and the starting age of the planning horizon is the same as the end age.
Because every machine has to be maintained at least once in the planning horizon, one y_{ik} can be fixed to 1. The cycle is then scheduled relative to this PM. Here, the PM at the first rank at the first machine is fixed (19). The last two expressions ensure the domains of the variables given in table 1.

3 Some problem properties

Given the model formulation and the problem description, some problem-specific properties can be observed. On the one hand, a lower bound on the objective value of the stochastic programming problem is presented if the decisions on production sequence and maintenance plan are fixed and, on the other hand, a lower bound is given for the case where just the sequence is fixed.

3.1 Lower bound for solutions with given sequencing and maintenance decisions

From the model, it can be seen that the solution highly depends on the consideration of the random failures, if decisions on production sequence and maintenance planning are given. In the single-machine literature, the expected-value problem is used to evaluate the random failures in every scenario (e.g., Cassady and Kutanoglu [2005]). The expected cycle time ct_i for a single-machine problem is obtained by replacing ξ^w_{ik} in equation (13) by its expectation $m(as_{ik}, ac_{ik})$ given by equation (4). Solving this problem leads to the same value as if every scenario is evaluated using a stochastic programming approach because only one critical machine exists. But for a multi-machine environment, the expected maximal cycle time $\mathbb{E}\left[\max_i(ct_i)\right]$ obtained by the stochastic programming approach differs from the expected-value problem evaluating the maximal expected cycle time $\max_i(\mathbb{E}\left[ct_i\right])$. In the latter, just one machine is responsible for the objective. Scenarios where another machine becomes the critical machine due to failures are omitted but can be captured with a stochastic programming approach.
In figure 2, three cases for the longest expected cycle time in a flowshop environment are illustrated. The arrows represent the critical machines. The first case shows the expected-value problem with the longest expected cycle time on the first machine. In the second case, a failure scenario with a failure on the first machine at the first job is shown. The critical machine stays the same. In the third case, however, the critical machine changes with a failure on the second machine at the first job. The contribution of these scenarios to the objective-function value are not considered in the expected-value problem.
By evaluating the maximal cycle time for every possible scenario, the expected maximal cycle time $\mathbb{E}\left[\max_i(ct_i)\right]$ is obtained. In every scenario, the critical machine only changes if

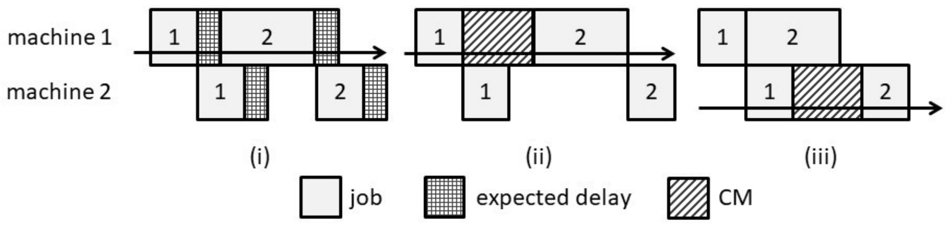

Figure 2: *Longest expected cycle time in multi-machine environment*

the cycle time on the contender is greater than that of the expected critical machine. This can only increase the resulting expectation. Hence, the inequality

$$\max_i(\mathbb{E}[ct_i]) \leq \mathbb{E}\left[\max_i(ct_i)\right] \tag{22}$$

holds for the objective-function values of both problems. The left-hand side is a lower bound for the expected maximal cycle time and is less time consuming to compute. This is because only one expected case needs to be evaluated here.
Given the assumption of multiple failures without any upper bound, there exist an infinite number of scenarios which need to be evaluated to obtain $\mathbb{E}[\max_i(ct_i)]$. To approximate this value, a simulation approach can be used that samples failure scenarios from an inverse Poisson process given the individual failure probabilities as discussed in Cui et al. [2018]. By approximating $\mathbb{E}[\max_i(ct_i)]$ in this way, not all scenarios are properly represented. Therefore, inequality (22) might not hold in some cases. This can lead to incorrect pruning as further discussed in section 4.3.

3.2 Lower bound for solutions with given sequencing decision

With the rule from the previous section, a lower bound on the expected maximal cycle time is obtained, if a sequence and maintenance plan is given. Now, a lower bound is discussed for the case that a sequence, but no maintenance plan is given. It estimates the quality of this sequence. For construction purposes only, the minimal possible expected processing times (including corrective maintenance)

$$EP_{ij}^{\min} = P_{ij} + T^{cm}\left(\frac{P_{ij}}{\eta_i}\right)^{\beta_i} \tag{23}$$

for jobs j on machine i and no PMs are taken into account. They can be calculated from the processing time and the expected delays if the prior age as_{ik} is zero. When solving the scheduling problem with these processing times, a lower bound for the expected maximal cycle time is obtained. To tighten this bound, one PM duration can be added to the machine's cycle time where the y_{i1} is fixed. This is due to the cyclic character of the schedule where no defined start of the planning horizon exists. However, the same cannot be done with the other machines. It is only known that every machine has to be maintained at least once but the position is unknown. It might be possible that the PM is not scheduled on the first rank. Therefore, this may be compensated with idle time during the production cycle. Because of the choice of $EP_{ij}^{\min}$, the objective-function value found with this consideration is always smaller than $\max_i(\mathbb{E}[ct_i])$ and forms a lower bound for the cycle time for a given sequence.

4 Solution approach

Both bounds from the previous section can now be embedded into a three-stage decomposition heuristic to solve the integrated problem. MONO is decomposed into three problems named the master problem (MP), the subproblem (SP) and the simulation (SIM). The MP uses the lower bound with given sequencing decision (see section 3.2). It takes only decisions on the production sequence while the SP decides on the maintenance plan for a fixed decision given by MP. The SP uses the lower bound with given sequence and maintenance decisions (see section 3.1) evaluating $\max_i(\mathbb{E}[ct_i])$. The simulation evaluates both decisions by approximating $\mathbb{E}[\max_i(ct_i)]$. This approximation is represented by the sampled confidence interval (CI). Two kinds of cuts are used to exchange information between the problems. First, cuts are used to exclude previously evaluated solutions from further consideration. Second, cuts on the objective-function value with the found solutions ensure that only reasonably good solutions are passed to the next stage.

4.1 Algorithm

The proposed approach can be described as illustrated in algorithm 1. The lines marked with (#) are optional lines where the presence depends on the used configuration in the computational study. *CI.mean* and *CI.ub* stand for the mean value of the CI and its upper bound, respectively.

Algorithm 1: Pseudo-code of algorithm

```
CI := InitalSolution;
while MP is not infeasible do
    $x$ := MP.solve();
    generate new SP;
    while SP is not infeasible do
        $y$ := SP.solve($x$);
        $CI'$ := SIM.solve($x, y$);
        if $CI'.mean < CI.mean$ then
            $CI := CI'$;
        end
        exclude $y$ by adding combinatorial Benders' cut to SP;
        set upper bound of SP to $CI.ub$ (#);
    end
    exclude $x$ by adding combinatorial Benders' cut to MP;
    set upper bound of MP to $CI.ub$ (#);
end
```

It starts with the generation of an initial solution to obtain a first hopefully high-quality upper bound. The idea is that the solution of the expected-value problem is close to the solution of the stochastic programming problem. Hence, only MP and SP of the proposed algorithm are solved to optimality and this solution is evaluated with SIM to find the starting incumbent solution. Then, the procedure starts again and a candidate solution for x_{jk} is generated by solving the MP that is handed over to SP. This generates a candidate solution for y_{ik} given x_{jk}. SIM evaluates the combination of both decisions and adjusts the approximated upper bound in case the new estimated value is better. Adding a combinatorial cut for the maintenance plan to the SP excludes this candidate from consideration and a new solution is generated in SP with respect to the current upper bound. To transfer information to the problem, a cut on the objective-function value of SP is added. Note here that this cut is optional to enable or disable the utilization of the lower bound. The evaluation of different maintenance plans for a given sequence terminates when every possible solution is excluded and the model becomes infeasible. Then, a combinatorial cut on the evaluated sequence and one objective-value cut to respect the upper bound is added to the MP. The objective-value cut is here also optional. A new candidate solution is generated until no feasible sequence can be found. The algorithm terminates at infeasibility and returns the incumbent solution.

4.2 Models

The MP, expressions (24) to (35), utilizes the lower bound from section 3.2 and consists of a modified version of the constraints (5) to (15) from MONO. The minimal expected processing times are given as data. They include P_{ij} and the minimal CM times given a starting age of zero. Because the PM activities are omitted (except that from the fixed machine), ps_{ik} and pc_{ik} are not needed in this model. The adapted scheduling constraints can be seen in expressions (30) to (33). The cycle times start directly with a production job and a PM duration is only added to the fixed machine (expressions (25) to (27)). Expressions (28) and (29) are the same as in MONO.

$$\min \quad ct_{\max} \tag{24}$$

$$ct_{\max} \geq s_{2|K|} - s_{11} + T_1^{pm} \tag{25}$$

$$ct_{\max} \geq s_{i+1|K|} - s_{i1} \qquad \forall \quad i \in I \setminus |I| \tag{26}$$

$$ct_{\max} \geq c_{|I||K|} - s_{|I|1} \tag{27}$$

$$\sum_{j \in J} x_{jk} = 1 \qquad \forall \quad k \in K \tag{28}$$

$$\sum_{k \in K} x_{jk} = 1 \qquad \forall \quad j \in J \tag{29}$$

$$s_{ik+1} \geq c_{ik} \qquad \forall \quad i \in I, k \in K \setminus |K| \tag{30}$$

$$c_{ik} = s_{ik} + \sum_{j \in J} x_{jk} EP_{ij}^{\min} \qquad \forall \quad i \in I, k \in K \tag{31}$$

$$s_{i+1k} \geq c_{ik} \qquad \forall \quad i \in I \setminus |I|, k \in K \tag{32}$$

$$s_{ik+1} \geq s_{i+1k} \qquad \forall \quad i \in I \setminus |I|, k \in K \setminus |K| \tag{33}$$

$$ct_{\max}, s_{ik}, c_{ik} \geq 0 \qquad \forall \quad i \in I, k \in K \tag{34}$$

$$x_{jk} \in \{0, 1\} \qquad \forall \quad j \in J, k \in K \tag{35}$$

The SP optimizes the maintenance decisions for a given sequence by evaluating the cycle time with the expected-value problem. Therefore, the lower bound from section 3.1 is used. The SP takes the objective and all constraints of MONO. Note here that x_{jk} is fixed in all constraints to the given sequence from the MP. Equation (13) is modified by replacing the realized number of failures ξ_{ik}^{w} by its expectation (4) depending on as_{ik} and ac_{ik}. This gives the model non-linear constraints resulting from $\beta_i > 1$. The scenario index as well as the expectation in the objective function can be omitted with this modification. To reformulate this problem to a MIP, a binary mapping from age to the expected number of failures is used.

Table 2: *Subproblem notations*

description	domain	type	meaning
B_i		index	set of mappings b at machine i
A_{ib}	(I, B_i)	data	possible age at machine i at mapping place b
F_{ib}	(I, B_i)	data	possible expected number of failures at machine i at mapping place b
λ_{ikb}^{as}	(I, K, B_i)	binary	assignment of starting age at machine i and rank k to mapping place b
λ_{ikb}^{ac}	(I, K, B_i)	binary	assignment of completion age at machine i and rank k to mapping place b

With the extended notation given in table 2, constraints (13) can be modified as (36) with the additional constraints (37) to (41). The mapping constraints (37) to (40) ensure that the ages prior and post the processing match to exactly one age in the predefined possible ages A_{ib}. These ages are calculated through the combination of all processing times per machine. With the binary mapping variables λ_{ikb}^{as} and λ_{ikb}^{ac} the age is mapped to the corresponding expected number of failures F_{ib}.

$$c_{ik} = s_{ik} + \sum_{j \in J} x_{jk} P_{ij} + T_i^{cm} \left(\sum_{b \in B_i} F_{ib} \lambda_{ikb}^{ac} - \sum_{b \in B_i} F_{ib} \lambda_{ikb}^{as} \right) \qquad \forall \quad i \in I, k \in K \tag{36}$$

$$as_{ik} = \sum_{b \in B_i} A_{ib} \lambda_{ikb}^{as} \qquad \forall \quad i \in I, k \in K \tag{37}$$

$$ac_{ik} = \sum_{b \in B_i} A_{ib} \lambda_{ikb}^{ac} \qquad \forall \quad i \in I, k \in K \tag{38}$$

$$\sum_{b \in B_i} \lambda_{ikb}^{as} = 1 \qquad \forall \quad i \in I, k \in K \tag{39}$$

$$\sum_{b \in B_i} \lambda_{ikb}^{ac} = 1 \qquad \forall \quad i \in I, k \in K \tag{40}$$

$$\lambda_{ikb}^{as}, \lambda_{ikb}^{ac} \in \{0, 1\} \qquad \forall \quad i \in I, k \in K, b \in B_i \tag{41}$$

SIM takes the objective and constraints from MONO and given decision on x_{jk} and y_{ik} to approximate $\mathbb{E}\left[\max_i(ct_i)\right]$. Here, ξ_{ik}^{w} takes its value from a sampled scenario which comes from an inverse Poisson process. The expected number of failures (4) can be calculated with given maintenance decision. Note that this and also the assignment to ranks can be done in a preprocessing step which reduces MONO to the objective function (5) with (6) and (7) as well as the scheduling constraints (10) to (15). To construct the CI, a Monte-Carlo approach is used as given in Birge and Louveaux [2011].

4.3 Exchange of information

To exchange information between the problems, different kinds of cuts are used. Combinatorial Benders' cuts are used to exclude found solutions from further consideration. The exchange of upper bound information enables the utilization of the lower bounds discussed above.
The combinatorial cut on the sequence is

$$\sum_{(j,k)\in R} x_{jk} \leq |R| - 1 \tag{42}$$

and that on the maintenance decision is

$$\sum_{(i,k)\in U} y_{ik} - \sum_{(i,k)\in \overline{U}} y_{ik} \leq |U| - 1 \tag{43}$$

as given in Codato and Fischetti [2006]. These cuts are weak and only exclude the particular solution.
The set R includes every (j,k)-pair for the binary decision variable x_{jk} which is 1 in the current solution and therefore $|R| = |J|$. Similarly, the set U includes the (i,k)-pairs where $y_{ik} = 1$ and a PM is scheduled. On the other hand the set $\overline{U}$ contains all (i,k)-pairs where no PM is scheduled and $y_{ik} = 0$ in the current solution. In contrast to the sequence cut where the current solution can be clearly defined by R because of the exact assignment constraints, both sets are needed to define the maintenance-plan part of the current solution.

Both cuts only exclude previously found solutions which leads to a full enumerative procedure over all feasible solutions. This is the case when no upper bound information is available to use the lower bounds. Therefore, the information of the sampled CI of the incumbent solution is used. To reduce incorrect pruning due to the approximative character of this bound the upper boundary CI_{ub} is used as the cut

$$ct_{\max} \leq CI_{ub} \tag{44}$$

on the objective value to ensure that every solution generated with one of the lower bound objective functions is smaller than the upper bound. $ct_{\max}$ stands here for the lower bound on the objective at the MP or SP. Therefore, the cuts restricting MP and SP are further called OC1 and OC2, respectively. The use of this upper bound information enables the lower bounds and reduces the number of feasible solutions of the individual problems. However, they can lead to a sub-optimal solution due to the quality of the estimation.

4.4 Implementation

The full algorithm is implemented in Python using Gurobi 9.1.0 as the MIP solver for all models. To accelerate the algorithm, cuts are implemented using lazy cuts as discussed in Fischetti et al. [2017]. This modern way of implementing cuts during the procedure speeds up the solution process. The cuts are added on-the-fly and there is no need to restart the search after new cuts are added. When an integral solution is found, solution-dependent cuts are added using a callback. Then, the algorithm searches for the next candidate. Hence, the search is done in a single tree and not an individual tree for each new cut.
SIM is called for every feasible solution that cannot be excluded and has the longest individual runtime. To decrease this, a simulation approach is used where the sample size is

adaptive and depends on the CI breadth. The breadth is interpreted here as a proxy for the estimation quality. If the breadth is small, the ongoing simulation can be stopped because it can be assumed that the quality is increased only slightly with additional samples. If it is large, more sampling is needed to tighten the CI. The simulation is stopped if the relative breadth reaches a given tolerance. Therefore, an increase of this or the confidence level of the CI would increase the quality of the estimation but also the runtime. To avoid premature, and, on the other hand, too late termination, an upper and lower limit for the number of samples are given. This accelerates the simulation if the deviation is small throughout the scenarios and bounds the worst-case runtime. The trade-off between quality and runtime can be seen for these bounds as well.

5 Computational study

In this section, the performance of the lower bounds and therefore the three-stage decomposition are evaluated. The proposed approach is tested in different configurations on an instance set and the results are compared to the benchmark algorithm without any use of upper bound information. All instances and results can be found in the supplementary material.

5.1 Data generation and design of experiments

Four different configurations ('conf') of the approach are tested to compare the performance of the lower bounds when information are exchanged via cuts. First, the benchmark approach (configuration 1) that uses no upper bound information and fully enumerates every feasible solution. Here, the information of both lower bounds cannot be used. The time limit for this configuration is set to six hours to receive good solutions also for larger instances. The goal is here to evaluate the solution quality of the other configurations. Configuration 2 uses only OC1 and configuration 3 only OC2 to avoid unnecessary simulations. Configuration 4 generates both at the procedure. The time limits of configurations 2-4 are set to one hour.
The configurations are compared by quality of solution which can be expressed as the comparison of the configuration solution to the best found solution among all configurations ('best') and whether they found the solution in the given time limit ('term') or not. In addition, the total runtime ('t-tot') and the relative time percentages the algorithm spends in the different sub parts are measured for MP ('rt-mp'), SP ('rt-sp') and SIM ('rt-sim'). The time to find the starting solution is not included in the total runtime. As a further competitive measure, every configuration that is fastest on an instance gets a point while the rest receive zero points. The average of these results is shown as 'win'. To obtain an impression of how many solutions and therefore simulations can be excluded, a performance measure for every configuration is provided. This measurement, in this work named the cut ratio, is measured by the number of feasible solutions found by the configuration divided by the number of possible feasible solutions. For the MP, there are $|J|!$ possible feasible solutions resulting from the permutation of all jobs. For every sequence found, a SP is solved which leads to a total of $|J|!2^{|J|-1}(2^{|J|}-1)^{|I|-1}$ feasible solution combinations. Included are the permutation of jobs and all combinations of y. The first term for y pertains to the first machine where the first y is fixed. The second term represents every other machine where it is ensured that an all zero y is infeasible. The cut ratio for the total problem ('cr-tot') is then expressed as one minus the total number of simulations divided by the number of total feasible solutions. The same way, the cut ratio for the MP ('cr-mp') is calculated. The

cut ratio of the SP ('cr-sp') takes the number of found solutions and the number of feasible solutions. This depends on the number of sequences excluded in the MP. To evaluate the benefit of the stochastic programming approach, the measurement 'Δ' is reported. Here, the initial solution, obtained by simulating the solution of the expected-value problem, is divided by the best solution found. Therefore, the percentage improvement of the solution can be seen.

The test set generated for this study consists of ten configurations with ten instances each. Flowshop environments for 2 and 3 machines are used and the number of jobs reaches from 4 to 8. The processing times are sampled from a discrete uniform distribution in the range of 1 to $\lceil \tau_i^* \rceil$ where τ_i^* is the analytically optimal interval between two PM as given in Cassady and Kutanoglu [2005] for a machine i. The equation

$$\tau_i^* = \eta_i \left[\frac{T_i^{pm}}{T_i^{cm}(\beta_i - 1)} \right]^{\frac{1}{\beta_i}} \tag{45}$$

can be obtained by maximizing the steady-state machine availability through differentiation and algebraic analysis given the machine parameter and hazard function. This ensures that the maximal processing time changes with different machine parameters. All machines are assumed to be different from a maintenance perspective and therefore $T_i^{pm} = [5, 10, 5]$, $T_i^{cm} = [15, 25, 10]$, $\beta_i = [2, 3, 2]$ and $\eta_i = [100, 100, 50]$. The parameter configurations of the first and the third machine are taken from the numerical example of Wang and Liu [2016] and the parameters of the second machine are taken from the solution analysis of Cassady and Kutanoglu [2003]. To estimate $\mathbb{E}[\max(ct)]$, a 95% confidence interval is used. There are at minimum 100 scenarios sampled from the simulation. It stops if a relative breadth of 5% or the maximum number is reached. To limit the growth in possible scenarios due to the instance size, the maximum number of samples is set to $|I| \times |J| \times 100$. For better comparison of the objective-function value and the cut ratio, the same random generator seed is used for the simulation at every configuration.

5.2 Results

The tests are run on an Intel(R) Core(TM) i5-6500 CPU machine at 3.2 GHz with 8 GB of RAM and the average results over the 10 instances are shown in tables 3 and 4. All solutions which exceed the time limit are excluded from the other performance measurements. The instances with 8 jobs for the 2-machine case (2I-8J) and those with 7 and 8 jobs for the 3-machine case are excluded because none of the configurations is able to solve them within the given time limit.
Over all instances and configurations, the solutions found with the stochastic programming approach is improved by approx. 2%. The results for different configurations may vary slightly due to multiple optimal solutions for the expected-value problem. The initial solution is found in an average time of 0.97 seconds for all instances and the CPU time ranges from 0.156 to 5.77 seconds taking the average time for the configurations in every configuration. A slight speed up of the solution process can be seen due to the OC1 cut (cut restricts the sequence generation with upper bound information). An evaluation of this speed up for larger instances may be a subject for future work.
The results reveal that only the configurations with OC2 (cut restricts maintenance decision in SP) are capable of solving the larger instances. Given the 'term' measurement, a first insight to the cut performance is that OC2 is better than OC1 because the configuration

Table 3: *Results for the 2-machine system*

$\|I\|$	$\|J\|$	conf	term	best	win	rt-mp	rt-sp	rt-sim	t-tot	cr-mp	cr-sp	cr-tot	Δ(%)
2	4	1	1	1	0	0.003	0.051	0.947	47.88	0	0	0	2.10
		2	1	1	0	0.003	0.050	0.946	36.08	0.263	0	0.263	2.09
		3	1	1	0.2	0.029	0.301	0.670	3.30	0	0.944	0.944	2.10
		4	1	1	0.8	0.026	0.257	0.717	3.18	0.263	0.913	0.942	2.09
	5	1	1	1	0	0.000	0.055	0.944	1027.88	0	0	0	2.11
		2	1	1	0	0.001	0.056	0.944	636.53	0.399	0	0.399	1.99
		3	1	1	0.2	0.013	0.386	0.601	28.12	0	0.978	0.978	2.11
		4	1	1	0.8	0.011	0.290	0.699	25.58	0.399	0.965	0.979	1.99
	6	3	1	1	0.4	0.007	0.474	0.519	349.99	0	0.992	0.992	2.08
		4	1	1	0.6	0.006	0.447	0.547	322.14	0.309	0.990	0.993	2.09
	7	3	0.6	1	0	0.009	0.644	0.347	2163.04	0	0.999	0.999	1.60
		4	0.7	1	1	0.004	0.548	0.448	1974.06	0.596	0.997	0.999	1.32

with only OC1 cannot solve the larger instances while configuration 3 with only the OC2 solves them.
This can also be seen from the cut ratio. It is every time higher for the SP than for the MP. OC1 excludes a higher amount of solutions per cut but cannot be used as frequently as OC2. For the smaller instances, from the results of configurations 2 and 3 the amount OC2 cuts away is round about twice the amount of OC1. The cut ratio increases over instance size. For the largest instances, a total cut ratio of nearly 99% suggests a good performance improvement through the cuts compared with the benchmark configuration. It also shows the computational effort to evaluate the remaining 1% of possible solutions. Comparing configuration 3 to configuration 4, it can be seen that the total cut ratio stays nearly the same but the distribution over the cuts differs.
The high overall cut ratio is reflected by the total solution time of all configurations. Comparing the ratio of average solution times of the benchmark configuration to every other configuration where it is possible (instances 2I-4J, 2I-5J and 3I-4J) a high speed up can be seen. Configurations 3 and 4, both containing OC2, are similar and approximately 13 to 40 times faster than the benchmark. Configuration 2 is only 1.3 to 1.6 times faster. Configuration 3 is the only one that solved an instance of 3I-6J. However, the average times for the larger instances (2I-7J and 3I-5J) display that Configuration 4 with both cuts performs better. The 'win' measurement also reflects this. Configuration 4 wins nearly every time more or as many instances of the data set.
The relative times spent in the different problems show a clear trend with increasing instance size and reflect also the evaluation of the cut ratio. The solution time of the MP stays relatively small at all times. The trend for the SP and SIM are opposite. The SIM time decreases while the cut ratio increases. The SP time increases in size. The cut ratio with OC1 is lower than with OC2. Therefore, the number of calls for the SP which cannot be cut away is higher. Furthermore, the solution time of the SP grows strongly with size. One reason for this may be the formulation with the binary mapping where the binary variables increase with the possible ages.
From the comparison of the configurations with information exchange (2-4) to the full enumerative benchmark configuration (1), it can be seen that in all cases where the approach terminates, the best solution is found. But the proposed approach does not guarantee to find this solution due to the upper bound approximation through simulation. It can be pruned

Table 4: *Results for the 3-machine system*

$\vert I\vert$	$\vert J\vert$	conf	term	best	win	rt-mp	rt-sp	rt-sim	t-tot	cr-mp	cr-sp	cr-tot	Δ(%)
3	4	1	1	1	0	0.000	0.040	0.959	780.99	0	0	0	1.93
		2	1	1	0	0.001	0.043	0.957	547.56	0.313	0	0.313	2.13
		3	1	1	0.4	0.008	0.071	0.921	59.58	0	0.921	0.921	1.93
		4	1	1	0.6	0.013	0.060	0.928	59.63	0.313	0.880	0.921	2.13
	5	3	1	1	0.5	0.001	0.134	0.865	841.44	0	0.982	0.982	2.06
		4	1	1	0.5	0.001	0.126	0.873	833.25	0.273	0.976	0.982	2.03
	6	3	0.1	1	1	0.001	0.427	0.572	2887.25	0	0.999	0.999	2.33

by both cuts. The use of a smaller tolerance combined with an increase of the upper simulation limit or a higher confidence of the CI can reduce the probability for this to happen but cannot guarantee the optimal solution.
Generally, the results show that the stochastic programming approach used here finds noticeably better solutions than the expected-value problem. The use of both lower bounds is sufficient to improve the performance of the benchmark algorithm, but also highlights the increasing solution time in the subproblem which can be tackled with a better formulation or solution approach. Further, the problem of sub-optimal solutions due to the approximation of the upper bound seems to be a drawback of this approach. Note that the results found here may not be representative due to small instances and sample sizes.

6 Conclusion

In this paper, a decomposition heuristic is presented to solve the IPSMP for a permutation flowshop environment with blocking, minimizing the expectation of the maximal cycle time. A sequence of jobs has to be scheduled. The machine's age increases while processing the jobs. With increasing age the machines are more likely to fail. The delays due to failure can be reduced by inserting preventive maintenance activities into the schedule. Two main properties of the problem are presented which give rise to the construction of lower bounds. This bounds are used in combination with combinatorial Benders' cuts to exclude solutions from the search procedure. The approach consists of three problems: the master and subproblem as well as a simulation to approximate the expectation of the maximal cycle time of a solution. For the simulation, further methods to accelerate the procedure are discussed. A computational study including 100 instances is used to test the performance improvements with used information. A benchmark approach that enumerates and evaluates all possible solutions is compared with other approaches using every other combination of the proposed cuts. The study shows that the cuts are able to highly improve the performance of the approach. This proposed mathematical programming approach thus fill the gap well between full enumerative and meta-heuristic approaches found in the literature.
For further research, the solution approach of the subproblem can be revised to find a better approach than the one presented. Exact branch-and-bound procedures or more heuristic approaches could be used to trade-off solution time and quality. Also different solution approaches for the simulation like Quasi-Monte-Carlo approaches can be used to either find better quality estimation in the same time or comparable quality estimations in less time. Both can be used to fine-tune the algorithm in terms of quality and runtime.

References

Aggoune, R. [2004]. Minimizing the makespan for the flow shop scheduling problem with availability constraints, *European Journal of Operational Research* **153**(3): 534–543.

Aggoune, R. and Portmann, M.-C. [2006]. Flow shop scheduling problem with limited machine availability: A heuristic approach, *International Journal of Production Economics* **99**(1-2): 4–15.

Allaoui, H., Lamouri, S., Artiba, A. and Aghezzaf, E. [2008]. Simultaneously scheduling n jobs and the preventive maintenance on the two-machine flow shop to minimize the makespan, *International Journal of Production Economics* **112**(1): 161–167.

Birge, J. R. and Louveaux, F. [2011]. *Introduction to Stochastic Programming*, Springer New York, New York, NY.

Breit, J. [2004]. An improved approximation algorithm for two-machine flow shop scheduling with an availability constraint, *Information Processing Letters* **90**(6): 273–278.

Breit, J. [2006]. A polynomial-time approximation scheme for the two-machine flow shop scheduling problem with an availability constraint, *Computers & Operations Research* **33**(8): 2143–2153.

Cassady, C. R. and Kutanoglu, E. [2003]. Minimizing Job Tardiness Using Integrated Preventive Maintenance Planning and Production Scheduling, *IIE Transactions* **35**(6): 503–513.

Cassady, C. R. and Kutanoglu, E. [2005]. Integrating Preventive Maintenance Planning and Production Scheduling for a Single Machine, *IEEE Transactions on Reliability* **54**(2): 304–309.

Cheng, T. and Wang, G. [2000]. An improved heuristic for two-machine flowshop scheduling with an availability constraint, *Operations Research Letters* **26**(5): 223–229.

Codato, G. and Fischetti, M. [2006]. Combinatorial Benders' Cuts for Mixed-Integer Linear Programming, *Operations Research* **54**(4): 756–766.

Cui, W. [2020]. Approximate Approach to Deal with the Uncertainty in Integrated Production Scheduling and Maintenance Planning, *Journal of Shanghai Jiaotong University (Science)* **25**(1): 106–117.

Cui, W., Lu, Z., Li, C. and Han, X. [2018]. A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops, *Computers & Industrial Engineering* **115**: 342–353.

Fischetti, M., Ljubić, I. and Sinnl, M. [2017]. Redesigning Benders Decomposition for Large-Scale Facility Location, *Management Science* **63**(7): 2146–2162.

Hadda, H. [2012]. A polynomial-time approximation scheme for the two machine flow shop problem with several availability constraints, *Optimization Letters* **6**(3): 559–569.

Hadda, H., Dridi, N. and Hajri-Gabouj, S. [2010]. An improved heuristic for two-machine flow shop scheduling with an availability constraint and nonresumable jobs, *4OR* **8**(1): 87–99.

Hadidi, L. A., Turki, U. M. A. and Rahim, A. [2012a]. Integrated models in production planning and scheduling, maintenance and quality: a review, *International Journal of Industrial and Systems Engineering* **10**(1): 21.

Hadidi, L. A., Turki, U. M. A. and Rahim, M. A. [2012b]. Joint job scheduling and preventive maintenance on a single machine, *International Journal of Operational Research* **13**(2): 174.

Hnaien, F., Yalaoui, F. and Mhadhbi, A. [2015]. Makespan minimization on a two-machine flowshop with an availability constraint on the first machine, *International Journal of Production Economics* **164**: 95–104.

Khatami, M. and Zegordi, S. H. [2017]. Coordinative production and maintenance scheduling problem with flexible maintenance time intervals, *Journal of Intelligent Manufacturing* **28**(4): 857–867.

Kubiak, W., Błażewicz, J., Formanowicz, P., Breit, J. and Schmidt, G. [2002]. Two-machine flow shops with limited machine availability, *European Journal of Operational Research* **136**(3): 528–540.

Kubzin, M. A. and Strusevich, V. A. [2005]. Two-machine flow shop no-wait scheduling with machine maintenance, *4OR* **3**(4): 303–313.

Kubzin, M. A. and Strusevich, V. A. [2006]. Planning Machine Maintenance in Two-Machine Shop Scheduling, *Operations Research* **54**(4): 789–800.

Lee, C.-Y. [1997]. Minimizing the makespan in the two-machine flowshop scheduling problem with an availability constraint, *Operations Research Letters* **20**(3): 129–139.

Lee, C.-Y. [1999]. Two-machine flowshop scheduling with availability constraints, *European Journal of Operational Research* **114**(2): 420–429.

Ma, Y., Chu, C. and Zuo, C. [2010]. A survey of scheduling with deterministic machine availability constraints, *Computers & Industrial Engineering* **58**(2): 199–211.

Ng, C. T. and Kovalyov, M. Y. [2004]. An FPTAS for scheduling a two-machine flowshop with one unavailability interval, *Naval Research Logistics* **51**(3): 307–315.

Rahmaniani, R., Crainic, T. G., Gendreau, M. and Rei, W. [2017]. The Benders decomposition algorithm: A literature review, *European Journal of Operational Research* **259**(3): 801–817.

Sadiqi, A., El Abbassi, I., El Barkany, A. and El Biyaali, A. [2018]. Joint Scheduling of Jobs and Variable Maintenance Activities in the Flowshop Sequencing Problems: Review, Classification and Opportunities, *International Journal of Engineering Research in Africa* **39**: 170–190.

Sanlaville, E. and Schmidt, G. [1998]. Machine scheduling with availability constraints, *Acta Informatica* **35**(9): 795–811.

Schmidt, G. [2000]. Scheduling with limited machine availability, *European Journal of Operational Research* **121**(1): 1–15.

Seif, J., Dehghanimohammadabadi, M. and Yu, A. J. [2019]. Integrated preventive maintenance and flow shop scheduling under uncertainty, *Flexible Services and Manufacturing Journal* **153**(3): 534.

Shoaardebili, N. and Fattahi, P. [2015]. Multi-objective meta-heuristics to solve three-stage assembly flow shop scheduling problem with machine availability constraints, *International Journal of Production Research* **53**(3): 944–968.

Sortrakul, N. and Cassady, C. R. [2007]. Genetic algorithms for total weighted expected tardiness integrated preventive maintenance planning and production scheduling for a single machine, *Journal of Quality in Maintenance Engineering* **13**(1): 49–61.

Vahedi-Nouri, B., Fattahi, P., Tavakkoli-Moghaddam, R. and Ramezanian, R. [2014]. A general flow shop scheduling problem with consideration of position-based learning effect and multiple availability constraints, *The International Journal of Advanced Manufacturing Technology* **73**(5-8): 601–611.

Wang, H. [2002]. A survey of maintenance policies of deteriorating systems, *European Journal of Operational Research* **139**(3): 469–489.

Wang, S. [2013]. Bi-objective optimisation for integrated scheduling of single machine with setup times and preventive maintenance planning, *International Journal of Production Research* **51**(12): 3719–3733.

Wang, S. and Liu, M. [2013]. A branch and bound algorithm for single-machine production scheduling integrated with preventive maintenance planning, *International Journal of Production Research* **51**(3): 847–868.

Wang, S. and Liu, M. [2016]. Two-machine flow shop scheduling integrated with preventive maintenance planning, *International Journal of Systems Science* **47**(3): 672–690.

Yulan, J., Zuhua, J. and Wenrui, H. [2008]. Multi-objective integrated optimization research on preventive maintenance planning and production scheduling for a single machine, *The International Journal of Advanced Manufacturing Technology* **39**(9-10): 954–964.

Chapter 6

Article 5: Mixed-integer formulations for the integrated production-scheduling and maintenance-planning problem in a flowshop

Mixed-integer formulations for the integrated production-scheduling and maintenance-planning problem in a flowshop

Sven Pries

Abstract

In this paper, different mixed-integer programming formulations are discussed to solve the integrated production-scheduling and maintenance-planning problem in a permutation flowshop. It is assumed that while jobs are processed, random failures can occur. These failures delay the completion and can be reduced by scheduling maintenance activities. The objective is to minimize the makespan for the expected number of failures. To cope with the non-linearities in the model, a binary-mapping and a constraint-based formulation are presented. The latter gives rise to a linear formulation for the quadratic case and an iterative approach for the general non-linear case. The performances of the constraint-based approaches are evaluated on a computational study and compared to the binary-mapping approach and a genetic algorithm. The results show that the constraint-based formulations are capable of solving small-size instances to optimality while competing with the frequently used meta-heuristic approaches on large-size instances.

Keywords: scheduling, maintenance, flowshop, mixed-integer programming, random failures

1 Introduction

The assumption of continuously available production resources (e.g., machines) is common in scheduling literature [Schmidt, 2000]. Hereby, realistic settings like non-availability intervals due to planned preventive maintenance (PM), corrective maintenance (CM) caused by random failures, or even lunch breaks are neglected. Decision support systems which incorporate these features into the schedules can benefit from this higher realism. Further, if not only the presence of these intervals but also the ability to plan them is incorporated, the systems can benefit from the interaction of the decisions. In the case of planned preventive maintenance combined with the scheduling of the production, the problem can be called the integrated production-scheduling and maintenance-planning problem (IPSMP). Here, maintenance activities are either needed to be scheduled as a necessity to maintain production or as an activity that is able to increase the production performance. Following the work of Cui et al. [2018], the problem can be subdivided into the deterministic and the stochastic problem. In the former, either PMs should be scheduled in given time limits or they can reduce degradation, for example, of job processing times. In the second, delays due to CMs caused by random failures are observed. These depend on the machine's condition and can be improved by the use of PMs.

The deterministic case can be further subdivided into the fixed and the flexible case. At the scheduling with fixed non-availability constraints, the non-available intervals are known in advance in number, start and duration. Therefore, no decision can be taken on these intervals. Comprehensive surveys for this case can be found in Sanlaville and Schmidt [1998], Schmidt [2000] and Ma et al. [2010]. For the flowshop setting, works with one [Hnaien et al., 2015] and multiple [Vahedi-Nouri et al., 2014; Shoaardebili and Fattahi, 2015] non-available intervals can be distinguished. If decisions on the intervals are allowed, the flexible case is observed for which partial surveys can be found in Hadidi et al. [2012a] and Sadiqi et al. [2018]. The intervals can be scheduled in predefined time windows [Mosheiov et al., 2018], as a way to reduce deteriorating processing times [Bajestani and Beck, 2015] or with deteriorating duration itself [Kubzin and Strusevich, 2005, 2006].
For the stochastic case, the machine condition depletes with the production of jobs and is then more likely to fail. The machine condition is represented by the total operational time since the last maintenance or simply as the age of the machine. The failures cause delays due to the need for CMs. PMs restore the machine condition and limit the impact while causing delays on their own. The machine age is set back here to an age of zero. Cassady and Kutanoglu [2003] consider this case first. A minimal repair at failure is assumed here. This means that the machine's age does not change at the occurrence of a CM. Only the PM can set back the age to a state of as-good-as-new, the so-called perfect repair. They use full enumeration to minimize the total weighted expected tardiness on a single machine by simultaneously scheduling jobs and PMs. It is also assumed that a machine can fail only once while processing a job. This limits the possible failure scenarios. They extend their work in Cassady and Kutanoglu [2005] to the total weighted expected completion time. The objective can be analyzed using the expected number of failures instead of scenarios and incorporating multiple failures. Small instances are solved using full enumeration and a heuristic is proposed for larger ones. In Sortrakul and Cassady [2007], a genetic algorithm is proposed for the problem of Cassady and Kutanoglu [2003]. A multi-objective version of the single-machine problem is studied in Yulan et al. [2008]. Five different objectives (among others tardiness and makespan) are evaluated. To solve the problem, a genetic algorithm is proposed. Pan et al. [2010] study the single-machine environment with the aim of minimizing the maximum weighted expected tardiness. They discuss the impact of the integrated model compared to an individual planning approach. Hadidi et al. [2012b] study the problem of Cassady and Kutanoglu [2005] and give a non-linear binary formulation to solve the problem. A joint model with production scheduling and predictive maintenance is given by Pan et al. [2012] for a single machine. They minimize the maximum expected tardiness. A bi-objective single-machine problem with total expected completion time and expected number of failures is given by Wang and Liu [2013]. He proposes a genetic algorithm to solve the problem. Further, Wang [2013] develop a branch-and-bound approach for the same problem minimizing the total expected weighted completion time. Abdelrahim and Vizvári [2017] discuss the problem with total expected completion time. They reduce a non-linear integer formulation of the problem to a 0-1 unconstrained optimization problem using problem-specific properties. Cui et al. [2020] study the solution robustness and quality robustness of the problem and propose a proactive joint model with random breakdowns. In Cui [2020], the integrated problem is extended to minimize the total energy costs as well as the makespan. A branch-and-bound approach and a hybrid non-dominant sorting genetic algorithm (NSGA-II) is proposed in this work.
Considering multi-stage environments, the impact of random failures is stronger because the machine coupling results in passing on delays. For the flowshop environment, Ruiz et al.

[2007] propose six adaptations of heuristics and meta-heuristics to minimize the makespan. Bajestani et al. [2014] model the problem differently using Markovian deteriorating machine conditions. They decompose the problem and solve it using a combination of Markov decision process and mixed-integer programming. Chen et al. [2015] assume imperfect repair at PMs and maximize the total profit. An immune clonal selection algorithm is proposed to solve the problem. In Seidgar et al. [2016], a bi-objective two-stage assembly flowshop problem is discussed. They formulate the problem with an availability approach and solve it using a non-dominant ranking genetic algorithm (NRGA). Xiao et al. [2016] propose a random-key genetic algorithm to minimize the total expected costs on a flowshop with group preventive maintenance. Wang and Liu [2016] suggest a genetic algorithm to solve the two-machine flowshop problem minimizing the makespan for the expected number of failures. Boudjelida [2017] studies the robustness of a flowshop environment minimizing a combination of makespan and a penalty for early or late maintenance. He proposes different meta-heuristics and also some hybridizations of them to solve this problem. Khatami and Zegordi [2017] solve the bi-objective problem with ant colony optimization. They minimize a combination of makespan and unavailability. Cui et al. [2018] study the multi-machine case and propose a two-loop algorithm for a bi-objective robustness problem. This algorithm includes a local search and a genetic algorithm for different decisions. Seif et al. [2019] suggest a two-stage stochastic mixed-integer program to the problem with expected-cost objective. A simulation-optimization approach with a genetic algorithm is used for larger instances.
As can be seen, less work is done for the flowshop environment in the stochastic case. Further, the used solution approaches focus more on meta-heuristic algorithms. The use of mathematical programming and in particular mixed-integer programming (MIP) formulations to the problem is neglected. This can be due to the formulation of uncertainty in the problem. A common way, among others, is to formulate the degradation process of the machine as a non-homogeneous Poisson process with a Weibull hazard function with shape parameter $\beta > 1$ as done in e.g., Cassady and Kutanoglu [2005], Wang and Liu [2016], or Cui et al. [2018]. This ensures an increasing failure rate due to increased age but gives the problem a non-linear character. This can be solved using a non-linear approach as done in Hadidi et al. [2012b], using meta-heuristics, or by linearization as an MIP. For the latter, no relevant approaches exist that utilize the good performance of modern MIP solvers for the problem. Another problem formulation is based on an availability approach. Here, a production-scheduling-specific objective is minimized by simultaneous consideration of minimizing the non-availability. Direct interactions of failures to the production scheduling are often neglected and only captured by the availability. These bi-objective approaches (see e.g., Seidgar et al. [2016] or Khatami and Zegordi [2017]) are solved typically with meta-heuristic algorithms.
To close these gaps, a MIP formulation to the flowshop problem of Wang and Liu [2016] is proposed in this work. Different ways to cope with the non-linearity are discussed. To show comparable performance of this approach to meta-heuristic algorithms, the performance is also compared to a genetic algorithm.
The rest of the paper is structured as follows. First, the problem is described in section 2 as a mixed-integer non-linear model. In section 3, ways to reformulate the non-linearity are discussed and an iterative cut approach is proposed for a Weibull shape parameter greater than two. In the computational study in section 4, the performance of the different MIP formulations is compared for small-size instances. Further, the proposed approach is compared to a genetic algorithm for larger instances. Section 5 concludes the paper.

2 Problem description

A set of jobs J needs to be scheduled on a set of machines I and the makespan for the expected number of failures should be minimized. The jobs need to visit every machine in a predefined order. At each machine i the job j has an integer processing time P_{ij}. For every machine, the jobs are assigned to a set of ranks K with $|K| = |J|$. This assignment of j to k determines the job sequence. Because a permutation flowshop is assumed, the sequence of the jobs being processed is the same for all machines. All jobs are available at time zero. A job can only be processed on the successor machine if it is completed on the current machine. An unlimited buffer is assumed and blocking is left unattended. To reduce the occurrence of random failures, PMs are scheduled in the time horizon which delay the job processing. These activities are assigned also to the ranks and scheduled prior to the job being processed on this rank. None of the jobs or PM intervals is allowed to interfere with each other. Hence, resumable jobs are not permitted. A PM activity is allowed to be executed as soon as the machine becomes idle. Random failures may delay the completion of jobs. Such a delay is modeled as an additional processing time resulting from the duration of a CM and the expected number of failures. The latter is machine-age dependent and can be controlled through the PM activities. Note here that only the expected-value problem and not a two-stage stochastic programming approach is considered for simplicity. Both can lead to different solutions due to the coupling in this environment.
In figure 1, a Gantt chart of a schedule for an example instance is given.

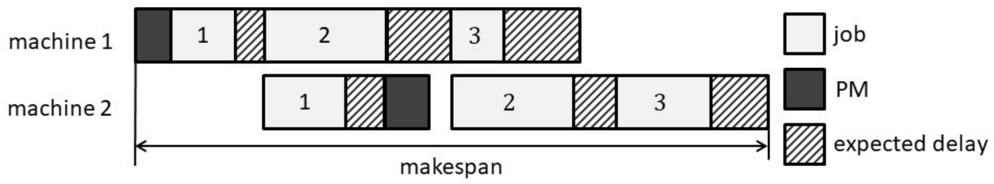

Figure 1: *Problem example*

Here, jobs 1 to 3 are processed on two machines. The dark rectangles represent scheduled PM activities. The time interval that is occupied by a job can be subdivided into a part with normal deterministic processing time and an expected-delay part (hashed rectangles). It can be seen that the PM scheduled on the second rank at the second machine is executed during idle time.
The interval that is occupied by a job assigned to a rank k at a machine i is defined by the starting time s_{ik} and completion time c_{ik}. The same can be done for the starting and completion times of the PM activities. Here, ps_{ik} defines the start and pc_{ik} the completion time, respectively. The binary variable

$$x_{jk} = \begin{cases} 1, & \text{if job } j \text{ is assigned to rank } k \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

is used to define the assignment of jobs to ranks.

The processing of jobs depletes the machine's condition and increases its age. The machine is then more likely to fail and delays due to CM activities occur more frequently. At failure, the machine is assumed to be minimally repaired, meaning that the CM sets the machine back to an operational state without changing the age. Given this assumption, every job processing interval at machine i is modeled as a non-homogeneous Poisson process with a

Weibull hazard function

$$z(t) = \frac{\beta_i}{\eta_i^{\beta_i}} t^{\beta_i - 1} \tag{2}$$

with scale parameter $\eta_i > 0$ and shape parameter $\beta_i > 1$ [Cassady and Kutanoglu, 2003]. The latter ensures the increase in failure rate as discussed previously. The process starts with an age prior (as_{ik}) and ends with an age post (ac_{ik}) the processing. The expected number of failures

$$\xi_{ik}(ac_{ik}, as_{ik}) = \left(\frac{ac_{ik}}{\eta_i}\right)^{\beta_i} - \left(\frac{as_{ik}}{\eta_i}\right)^{\beta_i} = \int_{as_{ik}}^{ac_{ik}} z(t)dt \tag{3}$$

can be calculated as a function of both ages and the given Weibull parameters [Cassady and Kutanoglu, 2005].
If the machine is down for failure, the completion of the job is delayed for a duration of T_i^{cm}. If it is down for a planned PM, an interval of T_i^{pm} is occupied. To plan a PM activity, the binary variable

$$y_{ik} = \begin{cases} 1, & \text{if a PM is scheduled at rank } k \text{ on machine } i \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

is used.
The PM times ps_{ik} and pc_{ik} exist at every rank, irrelevant of whether a PM is scheduled or not. A PM sets back the machine's age to zero. This is the only way to decrease the machine's age. To model the repair, a BigM formulation with a sufficient large value M_i^a is needed. M_i^a represents the maximal possible age for machine i. It can be calculated by the sum of all processing times on this machine and its starting age $a0_i$. The latter describes the age at the beginning of the planning horizon for each machine i. This is because the machines are not assumed to start the time horizon as new machines to better represent an ongoing planning and production. The combined notation can be found in table 1.
With this notation, a non-linear model can be obtained (expressions (5) to (17)). The non-linearity results from the function ξ_{ik} in equation (11). Other non-linear expressions as given in formulations from the literature (e.g., the age reset at PM as in [Cassady and Kutanoglu, 2005]) are already linearized.

$$\min \quad c_{|I||K|} \tag{5}$$

$$\sum_{j \in J} x_{jk} = 1 \qquad \forall \quad k \in K \tag{6}$$

$$\sum_{k \in K} x_{jk} = 1 \qquad \forall \quad j \in J \tag{7}$$

$$ps_{ik+1} \geq c_{ik} \qquad \forall \quad i \in I, k \in K \setminus |K| \tag{8}$$

$$pc_{ik} = ps_{ik} + T_i^{pm} y_{ik} \qquad \forall \quad i \in I, k \in K \tag{9}$$

$$s_{ik} \geq pc_{ik} \qquad \forall \quad i \in I, k \in K \tag{10}$$

$$c_{ik} = s_{ik} + T_i^{cm} \xi_{ik}(ac_{ik}, as_{ik}) + \sum_{j \in J} x_{jk} P_{ij} \qquad \forall \quad i \in I, k \in K \tag{11}$$

Table 1: *Notation*

description	domain	type	meaning
I		index	set of machines i
J		index	set of jobs j
K		index	set of ranks k
P_{ij}	(I, J)	data	processing time of job j on machine i
T_i^{pm}	(I)	data	time duration of PM activity on machine i
T_i^{cm}	(I)	data	time duration of CM activity on machine i
η_i	(I)	data	scale parameter of Weibull hazard function on machine i
β_i	(I)	data	shape parameter of Weibull hazard function on machine i
M_i^a	(I)	data	BigM representing the maximal age at machine i
$a0_i$	(I)	data	age of machine i at the beginning of the planning horizon
x_{jk}	(J, K)	binary	assignment of job j to rank k
s_{ik}	(I, K)	$\mathbb{R}^+$	starting time of rank k on machine i
c_{ik}	(I, K)	$\mathbb{R}^+$	completion time of rank k on machine i
ps_{ik}	(I, K)	$\mathbb{R}^+$	starting time of PM at rank k on machine i
pc_{ik}	(I, K)	$\mathbb{R}^+$	completion time of PM at rank k on machine i
y_{ik}	(I, K)	binary	PM decision for rank k on machine i
as_{ik}	(I, K)	$\mathbb{R}^+$	age before processing job on rank k on machine i
ac_{ik}	(I, K)	$\mathbb{R}^+$	age after processing job on rank k on machine i
ξ_{ik}	(I, K)	function	expected number of failures for rank k on machine i

$$s_{i+1k} \geq c_{ik} \qquad \forall \quad i \in I \setminus |I|, k \in K \tag{12}$$

$$ac_{ik} = as_{ik} + \sum_{j \in J} x_{jk} P_{ij} \qquad \forall \quad i \in I, k \in K \tag{13}$$

$$as_{i1} \geq a0_i - y_{i1} M_i^a \qquad \forall \quad i \in I \tag{14}$$

$$as_{ik+1} \geq ac_{ik} - y_{ik+1} M_i^a \qquad \forall \quad i \in I, k \in K \setminus |K| \tag{15}$$

$$s_{ik}, c_{ik}, ps_{ik}, pc_{ik}, as_{ik}, ac_{ik} \geq 0 \qquad \forall \quad i \in I, k \in K \tag{16}$$

$$x_{jk}, y_{ik} \in \{0, 1\} \qquad \forall \quad i \in I, j \in J, k \in K \tag{17}$$

The objective is to minimize the makespan for the expected number of failures considering the expected delays due to failures. This is the completion time of the last rank on the last machine (5). Equations (6) and (7) ensure that every job is assigned to exactly one rank and each rank is assigned to exactly one job. Expressions (8) to (12) ensure the non-interfering scheduling and the precedence constraints in the planning horizon. A PM activity can only start on a machine if the job on the same machine at the previous rank is completed (8). Equation (9) ensures that the PM duration is only added to the starting time if a PM activity is scheduled. The job processing can only start if the potential PM interval at the same rank is finished (10). The completion time of a job is given by expression (11). It depends on the starting time, the CM duration combined with the expected number of failures given

the machine ages and the deterministic processing time of the job assigned to this rank. Inequality (12) ensures that a job can only start on the successor machine if it is finished on the current machine. The age relationships are ensured by expressions (13) to (15). The age at a rank increases proportionally to the assigned job processing time while the job is being processed (13). Inequality (14) ensures the starting condition of the machines while (15) ensures that the age is set back to zero if a PM is scheduled. The last two expressions define the decision variable domains given in table 1.
To reformulate equation (11), different approaches can be used. Two of them are discussed in the following section for the special case of shape parameter $\beta_i = 2$ and for the general case of $\beta_i > 2$. For the latter, also an iterative approach is proposed to accelerate the solution process.

3 Solution approaches

3.1 Binary-mapping formulation

The non-linearity in equation (11) results from the function ξ_{ik} where ac_{ik} and as_{ik} are powered by β_i. A binary mapping that maps the particular age to the corresponding expected number of failures can be used for both of them. All possible age-failure pairs need to be given as data to perform this mapping.
The ages are determined by the processing times and must therefore be a combination of them. Other ages do not need to be observed. The possible ages at a specific machine are calculated using every combination of the processing times given at this machine. Further, both starting conditions (starting with $as_{i1} = 0$ after a PM or starting with $as_{i1} = a0_i$ without a PM) needs to be considered in this calculation. The set of these indexed possible ages b is denoted by Ω_i for every machine i. For the mapping of as_{ik}, the age of zero must be included and the combination of all jobs can be left unattended. From the possible ages given as the data A_{ib}, every possible expected number of failures F_{ib} can be calculated with the given Weibull parameters. With the binary-mapping variables λ^{ac}_{ikb} and λ^{as}_{ikb}, the age values are compared with A_{ib} and mapped to the expected number of failures.
Hence, equation (11) can be rewritten as

$$c_{ik} = s_{ik} + T^{cm}_i \left(\sum_{b \in \Omega_i} F_{ib} \lambda^{ac}_{ikb} - \sum_{b \in \Omega_i} F_{ib} \lambda^{as}_{ikb} \right) + \sum_{j \in J} x_{jk} P_{ij} \qquad \forall \quad i \in I, k \in K \tag{18}$$

with additional constraints

$$as_{ik} = \sum_{b \in \Omega_i} A_{ib} \lambda^{as}_{ikb} \qquad \forall \quad i \in I, k \in K \tag{19}$$

$$ac_{ik} = \sum_{b \in \Omega_i} A_{ib} \lambda^{ac}_{ikb} \qquad \forall \quad i \in I, k \in K \tag{20}$$

$$\sum_{b \in \Omega_i} \lambda^{as}_{ikb} = 1 \qquad \forall \quad i \in I, k \in K \tag{21}$$

$$\sum_{b \in \Omega_i} \lambda^{ac}_{ikb} = 1 \qquad \forall \quad i \in I, k \in K \tag{22}$$

$$\lambda^{as}_{ikb}, \lambda^{ac}_{ikb} \in \{0, 1\} \qquad \forall \quad i \in I, k \in K, b \in \Omega_i \tag{23}$$

to ensure the mapping. Expressions (19) and (20) represent the comparison of the ages to A_{ib} while expressions (21) and (22) ensure that only one age is selected.
This formulation can be used for every value of β_i but can grow large when the number of possible ages increases. These depend on the number of jobs and their processing times. Further, the preprocessing phase to calculate the data of A_{ib} and F_{ib} can be time consuming with increasing problem size. A non-exact approach to limit this growth would be to use piecewise linear functions to approximate the expected number of failures, but is neglected in this work.

3.2 Constraint-based formulation

In the previous formulation, multiple new mapping variables are included in the formulation. Now, a way to formulate the problem using only constraints and no additional variables is discussed.
Both ages can be combined to express the expected number of failures ξ_{ik} without being mapped separately. Note that ξ_{ik} is used here as a variable and not as a function as given previously. The model tends to decrease ξ_{ik} as much as possible. Hence, the inequality

$$\xi_{ik} \geq \left(\frac{ac_{ik}}{\eta_i}\right)^{\beta_i} - \left(\frac{as_{ik}}{\eta_i}\right)^{\beta_i} \qquad \forall \quad i \in I, k \in K \tag{24}$$

expresses the expected number of failures for every rank k on machine i. The denominators are combined and substituting $ac_{ik} = as_{ik} + \sum_{j \in J} x_{jk} P_{ij}$ results in

$$\xi_{ik} \geq \frac{1}{\eta_i^{\beta_i}} \left(\left(as_{ik} + \sum_{j \in J} x_{jk} P_{ij} \right)^{\beta_i} - as_{ik}^{\beta_i} \right) \qquad \forall \quad i \in I, k \in K. \tag{25}$$

Because exactly one job must be scheduled at a given rank, $\sum_{j \in J} x_{jk} P_{ij}$ can be substituted by P_{ij} and a constraint is added for every job j. If the specific job is assigned to this rank only the particular constraint should be used. This is formulated using a BigM formulation with the sufficient large number M_i^c that is discussed in more detail later. Note here that $\frac{1}{\eta_i^{\beta_i}}$ from inequality (25) is constant and can be moved to the reformulation of equation (11). To clarify this, $\bar{\xi}_{ik}$ is further used instead of ξ_{ik}. The reformulation is given by

$$c_{ik} = s_{ik} + \frac{T_i^{cm}}{\eta_i^{\beta_i}} \bar{\xi}_{ik} + \sum_{j \in J} x_{jk} P_{ij} \tag{26}$$

with the additional constraint

$$\bar{\xi}_{ik} \geq \left((as_{ik} + P_{ij})^{\beta_i} - as_{ik}^{\beta_i} \right) - M_i^c (1 - x_{jk}) \qquad \forall \quad i \in I, k \in K, j \in J \tag{27}$$

and a further variable domain

$$\bar{\xi}_{ik} \geq 0 \qquad \forall \quad i \in I, k \in K. \tag{28}$$

At $\beta_i = 2$ the quadratic terms with the decision variable as_{ik} cancels out and leave the linear inequality

$$\bar{\xi} \geq 2as_{ik} P_{ij} + P_{ij}^2 - M_i^c (1 - x_{jk}) \qquad \forall \quad i \in I, k \in K, j \in J \tag{29}$$

with P_{ij}^2 given as data. The value of M_i^c is calculated in this special case using the machine-specific largest processing time and the maximal value of as_{ik}. The latter is $a0_i$ combined with the sum of all processing times reduced by the shortest one.
For $\beta_i > 2$, inequality (27) remains non-linear but can be linearized using a lower linear approximation of the convex non-linear function. This is using the tangents on the possible ages. Hence, the formulation is exact with discrete ages. The set of possible ages b is also denoted here as Ω_i. For each possible age and each P_{ij}, the slope α_{ijb} and the intercept γ_{ijb} for every job j at machine i at the age indexed by b is calculated. This leads to the inequality

$$\bar{\xi} \geq \alpha_{ijb} as_{ik} + \gamma_{ijb} - M_i^c(1 - x_{jk}) \qquad \forall \quad i \in I, k \in K, j \in J, b \in \Omega_i \tag{30}$$

where M_i^c is calculated using the maximal as_{ik} and the slope as well as the intercept at the maximal age-processing-time combination.

3.3 Iterative cut approach

Two approaches to tackle the case of $\beta_i > 2$ are discussed in the previous section where both have their drawbacks with the need of knowing every possible age beforehand. The preprocessing of this data is time consuming when problems become large. However, the constraint-based formulation enables the possibility to give a lower bound on the expected number of failures when not all ages are indexed in Ω_i. Because not all possible ages at every machine are needed to form the optimal solution, only a subset Ω_i' is necessary. This idea can also be extended to the ranks. Hence, Ω_{ik}' is the reduced set of possible ages indices b for every machine i and rank k.
The set Ω_{ik}' is initialized only containing the age of zero and the maximal possible age of as_{ik}. An optimal solution to this simplified problem provides a lower bound to the solution value. Much more important, it indicates the ages that are needed to form this solution given the approximated expected number of failures. These values are added to Ω_{ik}' and further the corresponding constraints to the model. Therefore, the approximation becomes more precise. By solving the model again, a new solution is obtained. If no new ages, meaning ages that are not included in Ω_{ik}' over all ranks and machines, can be found, the algorithm terminates in the optimal solution. An upper bound to the solution can be obtained by solving the non-linear model with fixed decisions on assignment and maintenance plan. By doing so, the non-linear parts of the model are known and can be calculated beforehand leaving a linear model.
With this approach, the computational time of the preprocessing as well as the model size can be limited.

4 Computational study

4.1 Data generation and design of experiments

To evaluate the performances of the different MIP formulations and to show the performances of the MIP approaches compared to a meta-heuristic algorithm, two tests are performed. Both cases, with $\beta_i = 2$ and $\beta_i > 2$, are studied. First, the binary-mapping formulation (BinMap) is compared to the constraint-based formulations on instances with $|J| = 10$. The binary mapping is used for both cases of β_i. For the other formulation, the purely linear inequality is used for the quadratic case (CMap) and the iterative cut approach (ICA) for

larger shape parameter. It is focused here on the capability of finding the optimal solution in a given time limit. Second, a comparison of the constraint-based formulations (CMap and ICA) against a standard genetic algorithm (GA) is conducted. This test is run on instances with $|J| = 30$ jobs. The focus of this evaluation is the best found solution in a comparable runtime. Both instance-sizes are taken from the computational study of Wang and Liu [2016].
A standard GA with reinsertion, elitism, and linear ranking selection is used in the second test. The adjustment to the problem and the parameterization rests upon the approaches of Wang and Liu [2016] and Xiao et al. [2016]. The solution is encoded as a random-key representation for the sequence and a binary representation for the maintenance decision. It uses a one-point crossover with probability of 0.5 and a uniform mutation type with probability 0.1. The reinsertion is set to a parent portion of 0.3 with an elitism ratio of 0.01. The population contains 50 individuals.
For every given number of jobs, instances with 2 and 3 machines are generated. The processing times are sampled from a discrete uniform distribution $U[1, \lfloor \tau_i^* \rfloor]$ where

$$\tau_i^* = \eta_i \left[\frac{T_i^{pm}}{T_i^{cm}(\beta_i - 1)} \right]^{\frac{1}{\beta_i}} \tag{31}$$

is the optimal length between two consecutive PMs which can be obtained analytically by maximizing the steady-state availability as given in Cassady and Kutanoglu [2005]. However, this cannot usually be met and combinatorial approaches are necessary. Rounding down this value ensures that only jobs are generated that fit this value. This reflects the machine properties given by the maintenance parameters. These are used mainly from the computational study of Cassady and Kutanoglu [2003] with identical values $T_i^{pm} = 5$, $T_i^{cm} = 15$ and $\eta_i = 100$ for each machine i. Shape parameters of 2 and 3 are used and are also the same for all machines. The starting ages $a0_i$ are sampled similar to the processing times from $U[0, \lfloor \tau_i^* \rfloor]$. For every configuration of machine numbers and shape parameters, 25 instances are generated.

The time limit for the first test is set to 1800 seconds. If an approach terminates in the given time limit, this instance is marked as 'term'. The solution time 'sol-time' and time required for the preprocessing 'pre-time' is reported. For 'sol-time', also the coefficient of variation (CV) is presented. Further, 'gap' is 0 if the optimal solution is found and larger than 0 if the solution cannot be obtained (or proven to be optimal) in the given time limit. Note here that for BinMap and CMap the MIP gap, as reported by the solver, is used. For ICA, the relative gap between upper and lower bound of the iterative approach is used.
For the second test, the time limit is set to 600 seconds for the MIP approaches and the genetic algorithm. In addition to 'sol-time' and 'gap' (only for the MIP approaches), a 'wins' measurement is reported. Every time an approach finds the best solution, it receives a point. If both approaches find the best solution found in a $1e$-3 objective function value tolerance, both receive a point. To cope with the randomness of the genetic algorithm, 5 runs are performed for each instance. The indicator shows the relative wins of these runs compared to a single MIP run. The preprocessing times were neglected in this test due to their low significance. The measurement

$$\Delta = \frac{z - z^{best}}{z^{best}} \tag{32}$$

shows the relative objective function value difference of the found solution z to the best found z^{best} from both approaches. For the GA, the best solution from the 5 runs is used.

All approaches 'sol' are implemented in Python. Gurobi 9.5.0 is used for the MIP approaches. Tests were run on an Intel(R) Core(TM) i5-6500 CPU machine at 3.2 GHz with 8 GB of RAM. All instances and results can be found in the supplementary material.

4.2 Results

The evaluated results can be found in tables 2 and 3.

Table 2: *Results for the MIP-formulation comparison*

β_i	sol	$\|I\|$	term	pre-time (s)	sol-time (s) [CV]	gap (%)
2	BinMap	2	0.72	29.62	1065.40 [0.60]	0.2912
		3	0.16	43.55	1644.79 [0.24]	1.5316
	CMap	2	0.64	0.24	1245.41 [0.46]	0.5756
		3	0.96	0.05	341.73 [1.09]	0.0612
3	BinMap	2	0.28	30.41	1556.03 [0.32]	50.1036
		3	0.12	44.49	1729.05 [0.12]	54.6956
	ICA	2	0.08	0.20	1793.70 [0.02]	0.8172
		3	0.36	0.08	1526.71 [0.36]	0.1352

For the first test, it can be seen that both constraint-based formulations (CMap and ICA) outperform the binary-mapping formulation on the 3-machine configuration. An interesting observation is that both approaches perform worse on 2 machines than on 3 machines. The analysis of the results show that each instance, where the solution time differs much (instances which are solved fast by BinMap and not at all by the corresponding constraint-based formulation), have specific properties. The processing times of these instances show a small spread between highest and lowest value. Therefore, the processing times are all close by. It seems that the binary-mapping formulation can cope better with these instances while the constraint-based formulations cannot. On 2 machines, the BinMap performs better for $\beta_i = 2$ and nearly the same for $\beta_i = 3$. The differences in 'term' are also reflected by the average solution time. Here, ICA outperforms BinMap at 3 machines while CMap performs worse on 2 machines. Interesting to see is the larger preprocessing time of the binary-mapping approach because every possible age and the corresponding expected number of failures need to be calculated. The BinMap requires between 30 and 45 seconds while the preprocessing of the constraint-based approaches is less than 1 second. It remains unclear whether the lower preprocessing times at the larger number of machines is caused by the low scale or if there is another reason. However, no systematical issue could be found in the analysis. Further, the smaller MIP gaps of the constraint-based formulations even on configurations where 'term' indicates low performance are notable.
For the second test, the 'term' measurement is omitted because the MIP cannot prove the optimality of the solution in any instance. Therefore, the solution time is 600 seconds - the same - for all approaches. From the 'wins' measurement, it can be seen that CMap is able to win more instances than the GA. It seems to perform better on the 3-machine configuration than on the 2-machine. This could be influenced by the lower performance of the MIP approaches for 2 machines, but a lower performance of the GA on 3 machines is more reasonable. The MIP gaps of CMap are nearly 9% and the relative gap Δ to the best

Table 3: *Results for the comparison of MIP and GA*

β_i	sol	$\|I\|$	wins	gap (%)	Δ
2	CMap	2	0.936	8.6984	1.80e-05
		3	0.912	8.8340	4.32e-05
	GA	2	0.080	-	0.12e-03
		3	0.088	-	0.41e-03
3	ICA	2	0	5.9344	0.034
		3	0	6.3280	0.038
	GA	2	1	-	0
		3	1	-	0

solution remains small. At the GA, the latter is always worse. This is also reflected by the 'wins' measurement. However, the differences are small.
Comparing the GA to the ICA for $\beta_i > 2$, a different picture emerges as the GA wins all of the instances. The difference between the solutions (Δ) is larger for ICA compared to the CMap and zero for the GA. This means that the GA finds the best solution on every instance. The ICA only gets 'wins' points at some instances by finding the same solution. The gaps of the ICA is worse than that of CMap which reflects the findings of the first test. Overall, the constraint-based approaches for both settings of β_i outperform the binary-mapping approach on the 3-machine setting. On the 2-machine setting, BinMap and ICA are comparable while CMap is outperformed. Further, the constraint-based approaches show better results in the preprocessing. The CMap can win most instances over the GA. Although, ICA is outperformed by the the GA. However, the relatively low Δ of the ICA and the information about the solution quality ('gap') should not be disregarded here. The results show that the constraint-based formulations proposed in this work show well performance on solving smaller instances to optimality. The comparison to a standard genetic algorithm highlights the usability of mathematical programming for this kind of problem. One approach shows a good performance while the other, despite worse performance, provides useful insights into the solution quality.

5 Conclusion

The IPSMP discussed in this work is an integrated problem combining the decisions on the production scheduling with the maintenance planning. Both, production and maintenance, are occupying machine time in the planning horizon and must be balanced. While the processing of jobs depletes the machine's condition, maintenance activities restore it. As the condition deteriorates, the machine is more likely to fail randomly and requires corrective maintenance. This delays the completion time of the job. Since these failures are more serious when interconnected machines are assumed, a permutation flowshop environment is discussed in this work.
Mainly meta-heuristics and fewer mathematical programming approaches, especially MIP, have been used to solve this problem. This may be due to the non-linearity arising through the representation of the stochastic process. Two MIP approaches, a binary-mapping formulation and a constraint-based formulation, are discussed in this work to formulate the problem. The drawback of the first approach is that all possible ages need to be known in

advance. The second approach can overcome this problem by using an iterative approach for the general non-linear case and a simple linear formulation for the quadratic case.
Both formulations are evaluated on small-size instances for both cases. The constraint-based formulations show a good performance at this test. To show the further capability to compete with meta-heuristic algorithms on larger instances, a second computational study is presented. This shows a good performance for the linear formulation and worse performance for the iterative approach, but also indicates the usability of these kind of approaches for the IPSMP.
In the study also some interesting performance drawbacks are shown for the constraint-based formulations on the 2-machine case if the differences of the processing times is small. To evaluate the cause of this disadvantage and a way to overcome it, should be the focus of further research.

References

Abdelrahim, E. H. and Vizvári, B. [2017]. Simultaneous Scheduling of Production and Preventive Maintenance on a Single Machine, *Arabian Journal for Science and Engineering* **42**(7): 2867–2883.

Bajestani, M. A., Banjevic, D. and Beck, J. C. [2014]. Integrated maintenance planning and production scheduling with Markovian deteriorating machine conditions, *International Journal of Production Research* **52**(24): 7377–7400.

Bajestani, M. A. and Beck, J. C. [2015]. A two-stage coupled algorithm for an integrated maintenance planning and flowshop scheduling problem with deteriorating machines, *Journal of Scheduling* **18**(5): 471–486.

Boudjelida, A. [2017]. On the robustness of joint production and maintenance scheduling in presence of uncertainties, *Journal of Intelligent Manufacturing* **35**(5–6): 541.

Cassady, C. R. and Kutanoglu, E. [2003]. Minimizing Job Tardiness Using Integrated Preventive Maintenance Planning and Production Scheduling, *IIE Transactions* **35**(6): 503–513.

Cassady, C. R. and Kutanoglu, E. [2005]. Integrating Preventive Maintenance Planning and Production Scheduling for a Single Machine, *IEEE Transactions on Reliability* **54**(2): 304–309.

Chen, X., Xiao, L., Zhang, X., Xiao, W. and Li, J. [2015]. An integrated model of production scheduling and maintenance planning under imperfect preventive maintenance, *Eksploatacja i Niezawodnosc - Maintenance and Reliability* **17**(1): 70–79.

Cui, W. [2020]. Approximate Approach to Deal with the Uncertainty in Integrated Production Scheduling and Maintenance Planning, *Journal of Shanghai Jiaotong University (Science)* **25**(1): 106–117.

Cui, W., Lu, Z., Li, C. and Han, X. [2018]. A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops, *Computers & Industrial Engineering* **115**: 342–353.

Cui, W., Sun, H. and Xia, B. [2020]. Integrating production scheduling, maintenance planning and energy controlling for the sustainable manufacturing systems under TOU tariff, *Journal of the Operational Research Society* **71**(11): 1760–1779.

Hadidi, L. A., Turki, U. M. A. and Rahim, A. [2012a]. Integrated models in production planning and scheduling, maintenance and quality: a review, *International Journal of Industrial and Systems Engineering* **10**(1): 21.

Hadidi, L. A., Turki, U. M. A. and Rahim, M. A. [2012b]. Joint job scheduling and preventive maintenance on a single machine, *International Journal of Operational Research* **13**(2): 174.

Hnaien, F., Yalaoui, F. and Mhadhbi, A. [2015]. Makespan minimization on a two-machine flowshop with an availability constraint on the first machine, *International Journal of Production Economics* **164**: 95–104.

Khatami, M. and Zegordi, S. H. [2017]. Coordinative production and maintenance scheduling problem with flexible maintenance time intervals, *Journal of Intelligent Manufacturing* **28**(4): 857–867.

Kubzin, M. A. and Strusevich, V. A. [2005]. Two-machine flow shop no-wait scheduling with machine maintenance, *4OR* **3**(4): 303–313.

Kubzin, M. A. and Strusevich, V. A. [2006]. Planning Machine Maintenance in Two-Machine Shop Scheduling, *Operations Research* **54**(4): 789–800.

Ma, Y., Chu, C. and Zuo, C. [2010]. A survey of scheduling with deterministic machine availability constraints, *Computers & Industrial Engineering* **58**(2): 199–211.

Mosheiov, G., Sarig, A., Strusevich, V. A. and Mosheiff, J. [2018]. Two-machine flow shop and open shop scheduling problems with a single maintenance window, *European Journal of Operational Research* **271**(2): 388–400.

Pan, E., Liao, W. and Xi, L. [2010]. Single-machine-based production scheduling model integrated preventive maintenance planning, *The International Journal of Advanced Manufacturing Technology* **50**(1-4): 365–375.

Pan, E., Liao, W. and Xi, L. [2012]. A joint model of production scheduling and predictive maintenance for minimizing job tardiness, *The International Journal of Advanced Manufacturing Technology* **60**(9-12): 1049–1061.

Ruiz, R., Carlos García-Díaz, J. and Maroto, C. [2007]. Considering scheduling and preventive maintenance in the flowshop sequencing problem, *Computers & Operations Research* **34**(11): 3314–3330.

Sadiqi, A., El Abbassi, I., El Barkany, A. and El Biyaali, A. [2018]. Joint Scheduling of Jobs and Variable Maintenance Activities in the Flowshop Sequencing Problems: Review, Classification and Opportunities, *International Journal of Engineering Research in Africa* **39**: 170–190.

Sanlaville, E. and Schmidt, G. [1998]. Machine scheduling with availability constraints, *Acta Informatica* **35**(9): 795–811.

Schmidt, G. [2000]. Scheduling with limited machine availability, *European Journal of Operational Research* **121**(1): 1–15.

Seidgar, H., Zandieh, M. and Mahdavi, I. [2016]. Bi-objective optimization for integrating production and preventive maintenance scheduling in two-stage assembly flow shop problem, *Journal of Industrial and Production Engineering* **33**(6): 404–425.

Seif, J., Dehghanimohammadabadi, M. and Yu, A. J. [2019]. Integrated preventive maintenance and flow shop scheduling under uncertainty, *Flexible Services and Manufacturing Journal* **153**(3): 534.

Shoaardebili, N. and Fattahi, P. [2015]. Multi-objective meta-heuristics to solve three-stage assembly flow shop scheduling problem with machine availability constraints, *International Journal of Production Research* **53**(3): 944–968.

Sortrakul, N. and Cassady, C. R. [2007]. Genetic algorithms for total weighted expected tardiness integrated preventive maintenance planning and production scheduling for a single machine, *Journal of Quality in Maintenance Engineering* **13**(1): 49–61.

Vahedi-Nouri, B., Fattahi, P., Tavakkoli-Moghaddam, R. and Ramezanian, R. [2014]. A general flow shop scheduling problem with consideration of position-based learning effect and multiple availability constraints, *The International Journal of Advanced Manufacturing Technology* **73**(5-8): 601–611.

Wang, S. [2013]. Bi-objective optimisation for integrated scheduling of single machine with setup times and preventive maintenance planning, *International Journal of Production Research* **51**(12): 3719–3733.

Wang, S. and Liu, M. [2013]. A branch and bound algorithm for single-machine production scheduling integrated with preventive maintenance planning, *International Journal of Production Research* **51**(3): 847–868.

Wang, S. and Liu, M. [2016]. Two-machine flow shop scheduling integrated with preventive maintenance planning, *International Journal of Systems Science* **47**(3): 672–690.

Xiao, L., Song, S., Chen, X. and Coit, D. W. [2016]. Joint optimization of production scheduling and machine group preventive maintenance, *Reliability Engineering & System Safety* **146**: 68–78.

Yulan, J., Zuhua, J. and Wenrui, H. [2008]. Multi-objective integrated optimization research on preventive maintenance planning and production scheduling for a single machine, *The International Journal of Advanced Manufacturing Technology* **39**(9-10): 954–964.

Chapter 7

Appendix

7.1 Summary

This thesis is focused on the integrated production-scheduling and maintenance-planning problem with random failures. Production jobs need to be scheduled on one or several machine(s) while the processing of them depletes the machine's condition. With worsening condition, the machine is more likely to fail randomly and hence causing delays in the production. To reduce the possibility of failures, preventive maintenance activities can be scheduled throughout the production. These activities cause delays on their own. Hence, the random delays through failure and the planned delays through preventive maintenance must be balanced. This combined with the interrelations between production and the maintenance decision to fulfill a production-specific objective is the focus of this problem. The formulation used and the underlying assumptions are common in the literature. Given these assumptions, the problem is non-linear in character and little mathematical programming is used in the literature. Different approaches to this problem are proposed in this thesis. Mainly, mathematical programming and decomposition techniques are used to fill the revealed gaps.
For the single-machine version of the problem, two approaches are proposed. First, a branch-and-price algorithm is developed to solve the problem with expected-makespan objective. Different strategies to accelerate this algorithm are discussed and compared to the monolithic formulation of this problem. All show superior solving capabilities and outperform the monolithic approach clearly. Second, a branch-and-bound algorithm is proposed for the objective to minimize the total weighted expected completion time. Besides a mixed-integer program that can only solve small instances, this approach enhances a proposed branch-and-bound algorithm from the literature. An additional sequencing priority rule and problem-specific properties of the problem enables the solution of larger instances. Both can be utilized by the use of a general precedence formulation which is different to the mainly used rank-based formulation.
For the parallel-machine version of the problem, a decomposition approach is proposed decomposing the problem multiple times by use of a Benders' and a Dantzig-Wolfe decomposition. Through this approach the non-linearity is shifted to the subproblems and becomes more manageable. In addition to the used combinatorial cuts, also classical Benders' are derived from the good root relaxation of the Dantzig-Wolfe approach. Different configurations of the decomposition, the used cuts and their sequence of integration are compared with the monolithic model. All outperform this formulation. Further, the additional Dantzig-Wolfe decomposition and the classical Benders' cuts show an increase in performance.
For the flowshop version of the problem, a stochastic programming problem and an expected-value problem are proposed. A decomposition heuristic is developed for the stochastic programming. It uses problem-specific properties as lower bounds and decomposes the monolithic formulation in three problems. One for the sequencing decisions, one for the maintenance decision and a simulation to evaluate both decisions. The composed computational study focuses on the reduction of feasible solutions through the used lower bounds. For the expected-value version of the problem, an alternative mixed-integer formulation is proposed. A constraint-based formulation is developed to linearize the problem which is different to the binary mapping used in the previous works of this thesis. This can be also used in an iterative cutting approach. This formulation is compared to a binary mapping as well as a meta-heuristic approach and shows good performance.
All proposed approaches discussed in this thesis highlight the usability of mathematical programming and decomposition techniques for the given problem.

7.2 Kurzzusammenfassung

Im Rahmen dieser Arbeit wird das integrierte Produktionsplanungs- und Instandhaltungsplanungsproblem mit zufälligen Ausfällen behandelt. Produktionsaufträge müssen auf einer oder mehreren Maschinen eingeplant werden, während ihre Bearbeitung den Maschinenzustand verschlechtert. Je schlechter der Zustand der Maschine wird, desto wahrscheinlicher ist ein zufälliger Ausfall, der zu Verzögerungen in der Produktion führt. Um die Wahrscheinlichkeit von Ausfällen zu verringern, können während der gesamten Produktion vorbeugende Instandhaltungsaktivitäten eingeplant werden. Diese Aktivitäten verursachen ebenfalls Verzögerungen. Daher müssen die zufälligen Verzögerungen durch Ausfälle und die geplanten Verzögerungen durch vorbeugende Instandhaltungen abgewogen werden. Dies in Verbindung mit den Wechselbeziehungen zwischen der Produktions- und der Instandhaltungsentscheidung zur Erfüllung eines produktionsspezifischen Ziels steht im Mittelpunkt dieses Problems. Die verwendete Formulierung und die zugrunde liegenden Annahmen sind in der Literatur üblich. Angesichts dieser Annahmen ist das Problem nicht-linear, und in der Literatur wird selten mathematische Programmierung zur Lösung verwendet. In dieser Arbeit werden verschiedene Ansätze für das Problem vorgeschlagen. Hauptsächlich werden mathematische Programmierung und Dekompositionstechniken verwendet, um die aufgezeigten Lücken zu füllen.
Für die Ein-Maschinen-Version des Problems werden zwei Ansätze vorgeschlagen. Zunächst wird ein Branch-and-Price-Algorithmus dargestellt, um das Problem mit der Zielfunktion der erwarteten Zykluszeit zu lösen. Es werden verschiedene Strategien zur Beschleunigung dieses Algorithmus diskutiert und mit der monolithischen Formulierung des Problems verglichen. Alle zeigen eine überlegene Lösungsfähigkeit und übertreffen den monolithischen Ansatz deutlich. Zweitens wird ein Branch-and-Bound-Algorithmus zur Minimierung der Summe aller gewichteten erwarteten Fertigstellungszeiten vorgeschlagen. Neben einem gemischt-ganzzahligen Programm, das lediglich kleine Instanzen lösen kann, verbessert dieser Ansatz den in der Literatur vorgeschlagenen Branch-and-Bound-Algorithmus. Eine zusätzliche Prioritätsregel für die Reihenfolge und problemspezifische Eigenschaften des Problems ermöglichen die Lösung von größeren Instanzen. Beides kann durch die Verwendung einer allgemeinen Vorrangformulierung genutzt werden, die sich von der hauptsächlich verwendeten rang-basierten Formulierung unterscheidet.
Für die Parallelmaschinenversion des Problems wird ein Dekompositionsansatz vorgeschlagen, bei dem das Problem mit Hilfe einer Benders' und einer Dantzig-Wolfe-Dekomposition mehrfach zerlegt wird. Durch diesen Ansatz wird die Nichtlinearität auf die Teilprobleme verlagert und somit besser handhabbar. Zusätzlich zu den verwendeten kombinatorischen Schnitten werden auch klassische Benders' Schnitte aus der guten Relaxation des Dantzig-Wolfe-Ansatzes abgeleitet. Verschiedene Konfigurationen der Dekomposition, der verwendeten Schnitte und deren Integrationsreihenfolge werden mit dem monolithischen Modell verglichen. Alle schneiden besser ab als diese Formulierung und die zusätzliche Dantzig-Wolfe-Zerlegung sowie die klassischen Benders-Schnitte zeigen eine Leistungssteigerung.
Für die Flowshop-Version des Problems werden ein Ansatz für die stochastischen Programmierung und einer für das Erwartungswertproblem dargestellt. Für die stochastische Programmierung wird eine Dekompositionsheuristik entwickelt. Sie verwendet problemspezifische Eigenschaften als untere Schranken und zerlegt das Problem in ein Problem für die Reihenfolgeentscheidung, eines für die Instandhaltungsentscheidung und ein Simulationsmodell zur Bewertung beider Entscheidungen. Die zusammengestellte Rechenstudie zeigt die Reduktion der zulässigen Lösungen durch die verwendeten unteren Schranken. Für das

Erwartungswertproblem wird eine alternative gemischt-ganzzahlige Formulierung vorgeschlagen. Zur Linearisierung des Problems wird eine nebenbedingungs-basierte Formulierung verwendet. Diese unterscheidet sich von dem Binary Mapping Ansatz, welcher zum Teil in den früheren Artikeln dieser Arbeit genutzt wurde. Die vorgeschlagene Formulierung kann auch in einem iterativen Ansatz zur Schnittgenerierung verwendet werden. Diese Formulierung wird sowohl mit dem Binary Mapping als auch mit einem meta-heuristischen Ansatz verglichen und zeigt eine gute Leistung.
Alle vorgeschlagenen Ansätze in dieser Arbeit unterstreichen die Anwendbarkeit der mathematischen Programmierung und der Dekompositionstechniken für das gegebene Problem.

7.3 List of articles

Table 7.1: *Articles and authors*

title	authors
A branch-and-price algorithm for the integrated production scheduling and maintenance planning on a single machine	Sven Pries
A branch-and-bound approach for integrated production scheduling and maintenance planning on a single machine	Sven Pries
Decomposition approach for integrated production and maintenance scheduling on parallel machines	Sven Pries, Celso Gustavo Stall Sikora
Decomposition approach for integrated production scheduling and maintenance planning of a cyclic flowshop with random failures	Sven Pries
Mixed-integer formulations for the integrated production-scheduling and maintenance-planning problem in a flowshop	Sven Pries

7.4 The applicant's contribution

7.4.1 A branch-and-price algorithm for the integrated production scheduling and maintenance planning on a single machine

The article is single-authored.

7.4.2 A branch-and-bound approach for integrated production scheduling and maintenance planning on a single machine

The article is single-authored.

7.4.3 Decomposition approach for integrated production and maintenance scheduling on parallel machines

task	involved authors
Concept development	Pries
Benders' decomposition	Pries, Sikora
Dantzig-Wolfe decomposition	Pries
Implementation	Pries
Computational study	Pries
Manuscript	Pries, Sikora
Final examination	Pries, Sikora

7.4.4 Decomposition approach for integrated production scheduling and maintenance planning of a cyclic flowshop with random failures

The article is single-authored.

7.4.5 Mixed-integer formulations for the integrated production-scheduling and maintenance-planning problem in a flowshop

The article is single-authored.

7.5 Supplementary material

All supplementary material can be found at `http://doi.org/10.25592/uhhfdm.10253`

Eidesstattliche Versicherung:

Hiermit erkläre ich, Sven Henrik Pries, an Eides statt, dass ich die Dissertation mit dem Titel

„Selected topics on integrated production-scheduling and maintenance-planning problems“

selbständig - und bei einer Zusammenarbeit mit anderen Wissenschaftlern gemäß der beigefügten Darlegung - nach § 6 Abs. 4 der Promotionsordnung der Fakultät für Betriebswirtschaft vom 9. Juli 2014 verfasst und keine anderen als die von mir angegebenen Hilfsmittel benutzt habe. Die den herangezogenen Werken wörtlich oder sinngemäß entnommenen Stellen sind als solche gekennzeichnet.

Ich versichere, dass ich keine kommerzielle Promotionsberatung in Anspruch genommen habe und die Arbeit nicht schon in einem früheren Promotionsverfahren im In- oder Ausland angenommen oder als ungenügend beurteilt worden ist.

Lübeck, 08.11.2022 [illegible]

Ort/Datum Unterschrift