Simulation in Produktion
und Logistik 2019
Matthias Putz & Andreas Schlegel (Hrsg.)
Wissenschaftliche Scripten, Auerbach 2019

Simulation-Based Optimization Approach for Efficient Generation of Sequencing Heuristics for Solving the Stochastic Resource-Constrained Scheduling Problem

Ein simulationsbasierter Optimierungsansatz zur effizienten Generierung von Prioritätsregeln zur Lösung des stochastischen ressourcenbeschränkten Projektplanungsproblems

Mathias Kühn, Thorsten Schmidt, Michael Völker, TU Dresden, Dresden (Germany), {mathias.kuehn; thorsten.schmidt; michael.voelker}@tu-dresden.de

Abstract: This paper proposes a two-stage genetic algorithm based on a simulation-based optimisation approach for generating sequencing heuristics to solve the stochastic decentralised resource-constrained multi-project scheduling problem. Processing time deviations are considered as stochastic influences. The sequencing heuristics are a simple combination of different weighted job attributes, where the genetic algorithm specifies these weights. However, stochastic simulation and optimisation require a high simulation effort. A two-stage algorithm is proposed to tackle this problem. Since solutions of deterministic optimisation can also lead to good results in a stochastic environment, the first stage with less simulation effort is based on deterministic values. Afterwards, the algorithm selects promising deterministicly solutions to create a good initial solution for the second, more computational intensive stage of simulation with stochastic values. The strategy shows good performance compared to the results of randomly generated initial solutions.

1 Introduction

Individualisation as an industrial megatrend manifests itself in a greater diversity of variants (Schlick et al. 2017, p. 3). Particularly in the assembly of large individual products, which is characterised by human work, the associated low repetition rate leads to an increase in fuzzy process parameters. These increase the effort in production planning and control. In addition, stochastically influenced process times are inherent due to the fluctuating individual performance of human beings and will therefore continue to exist. Representative examples of the above-mentioned characteristics include the final assembly of customer-specific machine tools or complex conveyor systems, where several projects with individual objectives compete for resources. Abstracted, this problem is a stochastic resource-constrained multi-project scheduling problem (SRCMPSP), where the solution is for example a valid initial schedule. At present, either central Manufacturing Execution Systems (MES) with integrated Advanced Planning and Scheduling Systems (APS) or simple priority rules (SPR) are in use to control such requirements. SPR are often unsuitable in a stochastic environment and it is not possible to consider project-specific objectives. The requirements for using MES with APS are high data quality and real-time communication in order to be able to react to disturbances. This is often not the case in manual assembly. Resource-constrained planning problems with a high proportion of manual work are often characterised by large work packages from a time perspective. This means continuously recording the status in production is not possible, whereby new production plans may be immediately invalid which also indicates against the use of MES with APS. As an approach, production would have to run autonomously for a defined period, i.e. without monitoring and intervention by the central control system. The result of a process planning is therefore not a schedule, but must be an instruction for order processing, e.g. in the form of a scheduling heuristic, which is referred to below as a solution.

Researchers proposed many approaches for solving similar problems, such as job-shop problems differing from project planning problems in a lower network complexity. In the field of automated generation of scheduling heuristics, composite dispatching rules (CDR) are in the focus of interest. The approach of generating composite dispatching rules is adopted and evolved with fixed-length parametric representation by using a discret event simulation-based optimisation approach with a genetic algorithm as a hyper-heuristic for this special problem. Essential innovations are the handling of the problem class itself (SRCMPSP) and the efficient (with low simulation effort) generation of the CDR. The efficient generation of CDR becomes necessary because production control has to be readjusted more and more often due to shorter product life cycles. The necessary benchmark tests, e.g. versus SPR with different parameters, require a lot of simulation effort and therefore time. However, in reality, this time is not available and, in the worst case, the production control is not configured optimally. This requirement is further aggravated by the stochastic environment, which requires robust and therefore stochastic influences compensating CDR. Since rules trained on static, deterministic problems usually lead to worse results in a dynamic, stochastic environment, it is better to use stochastic, dynamic problems during training, which entails an even higher simulation effort (Branke et al. 2016). However, since deterministic solutions do not necessarily lead to bad solutions in a dynamic environment, they are suitable as initial solutions for stochastic optimisation. In this paper, a 2-stage strategy for generating initial solutions to save simulation effort is presented. The first stage consists of a low effort deterministic simulation and optimisation to explore the solution space. Copied solutions with good results from the first stage and randomly generated individuals to provide spread-off, form the initial solution for stochastic optimisation. The result of the stochastic optimisation is robust CDR by which a statistical parameter of the objective function value can be optimised under stochastic influences.

The structure of the paper is as follows: Chapter 2 describes the stochastic extensions to the considered resource-constrained multi-project scheduling problem and requirements of the project-specific pareto optimisation. Chapter 3 shows the state of the art on sequence heuristics. Chapter 4 presents the developed algorithm. Chapter 5 includes experiments to determine the saving of simulation effort and a comparison to SPR. Chapter 6 summarises the results.

2 Model: Extensions of the DRCMPSP

This paper consideres the decentralised resource-constrained multi-project scheduling problem (DRCMPSP) as described by Homberger (2007). This problem can be described briefly as follows: A set of $i \geq 2$ projects has to be planned simultaneously by self-interested decision makers. Each project has a set of jobs with precedence relations and arrival dates. The projects share a set of renewable resources. With the extension of stochastically processing times and the scope of single project optimisation (pareto-optimisation of each project), the problem can thus be described as a stochastic DRCMPSP (SDRCMPSP).

Stochastically job durations

The duration of a job $j \in \{1, \dots, |J|\}$ of project $i \in \{1, \dots, |I|\}$ is p_{jin}. p_{jin} is a random variable of a realised stochastic scenario $n \in \{1, \dots, |N|\}$ with a known distribution (expected processing time $E(p_{jin}) = \mu$ for all $n \in \{1, \dots, |N|\}$ and coefficient of variation var are given). The processing time must be $p_{jin} \geq 0$ (except dummy-jobs). The matrix P gathering all stochastic processing times $P = \left[\begin{bmatrix} p_{111} & \cdots & p_{1i1} \\ \vdots & \ddots & \vdots \\ p_{|J|11} & \cdots & p_{|J||I|1} \end{bmatrix}, \dots, \begin{bmatrix} p_{11|N|} & \cdots & p_{1i|N|} \\ \vdots & \ddots & \vdots \\ p_{|J|1|N|} & \cdots & p_{|J||I||N|} \end{bmatrix} \right]$. We focus on logarithmic normal distribution with a coefficient of variation $var_{lognorm} = 0.1 - 0.9$.

When jobs are scheduled, only the expected processing time μ and its coefficient of variation var are known.

Pareto-optimisation of project-oriented single objective functions:

In a stochastic environment, the mean of project delay M_PD_i of N stochastic scenarios is calculated by Equation 1.

$$M\_PD_i = \frac{1}{|N|} \sum_{n \in N} \left(\omega_{|J|in} - f_i \right) \tag{1}$$

where $\omega_{|J|in}$ is the scheduled finish time of project i and f_i is the determined finish time of project i.

The standard deviation of project delay M_PD_i of N stochastic scenarios is calculated by Equation 2.

$$STD\_PD_i = \sqrt{\frac{1}{|N|} \sum_{n \in N} \left(\omega_{|J|in} - f_i - M\_PD_i \right)^2} \tag{2}$$

Another objective value is the makespan, calculated by Equation 3.

$$MS_i = \left(\omega_{|J|in} - \alpha_{1i} \right) \tag{3}$$

where α_{1i} ist the start time of the first activity.

The corresponding statistical objective values to Equation 3 are calculated in the same way as Equation 1 and Equation 2 (STD_MS_i; M_STD_i).

The objective values MS_i and PD_i are not combined in an objective function together (two factors). Statistical parameters for each objective value are assigned individually ($STD\_$ or $M\_$) or in combination ($STD\_$ and $M\_$), so that in total six objective functions are considered.

An example for one of the six multi-objective functions taking the assumptions and constraints made into account is given in Equation 4:

$$minimize\{M\_PD_1, \dots, M\_PD_{|I|}, STD\_PD_1, \dots, STD\_PD_{|I|}\} \qquad (4)$$

3 State of the art

Many researchers have already been working on solving the stochastic scheduling problem. The basic concepts for solving these problems are reactive, stochastic, fuzzy and robust scheduling (Herroelen and Leus 2005). Considering the special domain of manual assembly as described in the introduction, stochastic scheduling without a baseline schedule is a good solution strategy for short-term production control. Thereby the processing sequence of a defined order pool is determined directly before the order release. For sequencing, scheduling policies, branch-and bound-procedures and heuristic procedures are mainly used. Heuristic procedures like scheduling heuristics such as CDR, are suitable for the considered problem domain because of low information requirements. A comprehensive review about CDR is given by Branke et al. (2016). In addition to the comparison of state-of-the-art approaches, a taxonomy is defined and guidelines for the design of CDR are proposed. Furthermore, challenges such as the computing time for CDR and the robustness of CDR are presented. Following conclusions are drawn for the presented work: While sequencing with CDR is comparatively fast, generating CDR with hyper-heuristics such as a genetic algorithm is often very computationally intensive. If the genetic algorithm is applied to a training set with stochastic variables, the simulation effort is further increased. A first possibility to reduce the computing effort is the representation of the CDR. The grammar-based tree representation (composition of individual components to a function on the basis of a parameterisable grammar) has in theory no fixed length and is suitable for larger computational budgets (Branke et al. 2015). Another variant is the parameter-based representation, which is a parameterisable priority function with defined format. The defined format means a fixed length and thus a calculable computational effort. Therewith a parameter-based representation is more suitable for the considered problem. Further methods to reduce the computing time refer to strategies of effective fitness evaluation and effective bloating control. The last-mentioned proposal concerns in particular methods for reducing and controlling the complexity of CDR. An example is the limitation of the tree-based genetic programming (Luke and Panait 2006). Other possibilities are the mathematical simplification of the evolved CDR (Wong and Zhang 2006). These methods are only conditionally applicable for parameter-based CDR, since the complexity does not increase with evolution. Promising approaches for the focused CDR can be found in the definition of stopping criteria (Baek and Yoon 2002). Another method to reduce the simulation effort is the generation of a good initial solution, which did not receive much attention (Jorapur et al. 2016). Compared to a randomly generated solution it can be assumed that the algorithm converges earlier and thus less simulation runs are required. The initial population of genetic algorithms for shop scheduling is predominantly generated randomly (Werner 2013, p. 11). An approach for improvement consists for example in the use of existing priority rules to form the initial solution (Omar et al. 2006). However, the research potential for generating promising initial solutions is an ongoing issue and is addressed in this paper. Even with the reduction of the computation time and thus simulation effort, the generation of CDR with hyper heuristics remains computationally intensive. For this reason, reaction to disturbances is very important. In contrast to individual concepts, which aim at a recalculation of the CDR in case of disturbances (Frazzon et al. 2018), the compensation of disturbances by CDR itself is in the core of the research. Therefore, the statistical significance and evaluation of objective values is emphasised. Another challenge and difference to previous work is the addressed problem class itself. While most papers consider job shop scheduling problems, this paper considers the more general project scheduling problem. For the realisation of an individual objective fulfilment, it is therefore necessary to assign an individual CDR to each project, whereby the complexity increases further.

4 Proposed concept of the two-stage genetic algorithm

The basic algorithm was presented in Schmidt et al. (2017). The idea is to weight different job attributes (e.g. due date, expexted processing time) in a sum function to calculate a priority which job has to be performed next. In summary, ten different job attributes where combined in this weighted sum approach (belongs to parametric presentation with fixed length) where each project is assigned its own weighted sum function. After a random initialisation of the weights, the genetic algorithm as the hyper-heuristic determines the weights of the CDR using the theory of simulation-based optimisation (Fig. 1). This means that the quality of each weighted sum function is evaluated with the help of discrete event simulation. Solving scheduling problems with discrete-event simulation is very common (Klemmt 2012, p. 57) and also used here as the solution approach with the simulation and optimisation framework PyScOp (Kühn et al. 2019). The algorithm stops when the stop criterion is reached, here for example a defined number of generations.

Considering stochastic influences in training data results in high simulation effort. For example, with a population size ps of 100 individuals and 100 generations gen (stop criteria) and 100 stochastic scenarios N, the simulation effort would amount to a total of 1,000,000 simulation runs ($ps \cdot gen \cdot N$). To reduce the effort and thus the time for generating the CDR, one objective is to generate optimised promising initial solutions. With optimised initial solutions, better solutions can be found with identical simulation effort than with randomly generated initial solutions. Furthermore, there is also the possibility to abort the simulation because solutions with the same quality are more likely to be found than with randomly generated initial solutions.

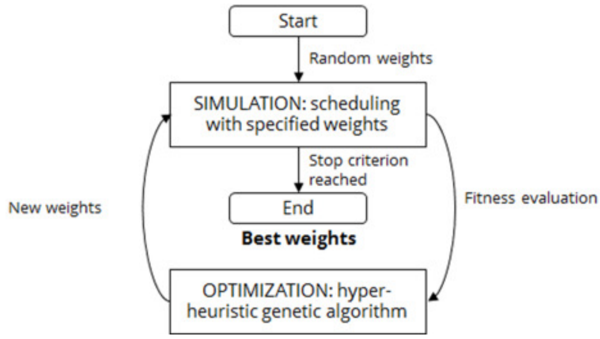

Figure 1: *Hyper-heuristic for the generation of sequencing heuristics (based on Branke et al., 2016)*

Numerous investigations were carried out comparing the results of deterministic solutions in a stochastic environment (Hildebrandt et al. 2010). Tendencies can be observed that deterministic results correlate with stochastic results. An example of correlation between deterministic and stochastic solutions is shown in Figure 2. In the example, both linear regressions of stochastic and deterministic results show positive slopes. Whether this correlation is in general valid or only for the investigated examples is another object of research. Even with a negative slope of the stochastic results, it can be assumed that among the best deterministic results, there are also good stochastic results. However, one cannot speak of a rule and therefore there are always several solutions to consider of the deterministic results. In conclusion, deterministic solutions may be well suited for creating the initial solution for stochastic optimisation. Proofing this hypothesis is part of the research.

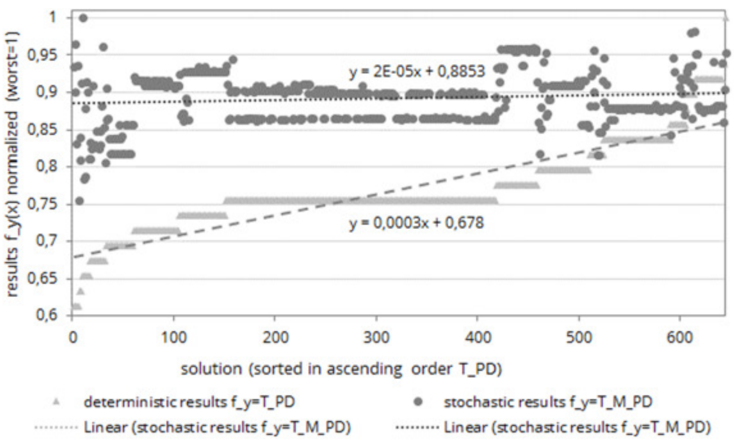

Figure 2: Correlation between deterministic and stochastic results

To improve generating CDR on training data with stochastic variables taking less simulation effort and better performance into account, the algorithm consists of two stages: A first deterministic stage, which means that the CDR are generated on a deterministic training set and a second stochastic stage, which means that based on the initial CDR further CDR are developed on a stochastic training set.

For selecting promising solutions, the parameter initial copy rate ICR is introduced. A copy rate of 0.2 means that the initial solution for stochastic optimisation consists of 20 % copied solutions from the first stage. The other 80 % are generated randomly. The best individuals with respect to the objective function value are copied.

Since there are more than one non-dominated solutions in the pareto front, we use the minimum sum of selected objective values as a tiebreaker.

The total sum of mean of e.g. project delay T_M_PD is calculated by (Eq. 5):

$$T\_M\_PD = \frac{1}{|I|}\sum_{i \in I} M\_PD_i \tag{5}$$

The best solution is indexed with $best$ $(e.g.\ T\_M\_PD_{best})$. Equation 5 is also valid for the considered objective functions (Eq. 2 and Eq. 3).

The algorithm was implemented in the PyScOp software framework (Kühn et al. 2019). Main extensions to the previous version are the stochastic simulation of processing times and thus the statistical evaluation of objective values.

5 Experiments

Benchmark problems from the MPSPLIB (multi project scheduling problem library) (Homberger 2019) were used to evaluate the concept.

5.1 Experiment design

The MPSPLIB is a library with 140 different instances differing in number of ressources, projects, jobs and network character. Solutions can be uploaded and checked for feasibility. The models are extended with the considered stochastic influences (Chapter 2). A 2-factor plan is chosen as the experimental plan. Each factor is assigned two levels and is fully factorially combined. The levels should represent a low level and a high level. This type of experimental design is chosen if many factors are to be investigated and experiments are very computation-intensive. The selected models of the MPSPLIB differ in the number of jobs $j = 30/120$, number of projects $i = 2/5$ and $iz = 2$ different instances (network character), so that a total of eight models M are considered. The distribution of the process time is considered with $var_{lognorm} = 0.1/0.9$. Furthermore, there are six objective functions f_y combinations. Thus, 96 factor combinations are possible in total (2 $var_{lognorm} \cdot 8\ M \cdot 6\ f_y = 96$). These factor combinations are combined with the initial copy rate $ICR = 0.0/0.2/0.4/0.6/0.8/1.0$ and also with the number of stochastic scenarios. We chose 100 stochastic scenarios N as the sample size (usual, see Freitag and Hildebrandt, 2016: $N = 50$; Pickardt et al., 2013: $N = 100$).

5.2 Comparison of the simulation effort

In order to evaluate whether a saving is possible using the presented 2-stage algorithm, the comparison of the simulation effort between the strategy for generating the initial solution and the randomised generation of the initial solution is necessary. The deterministic simulation effort SA_{det} for the generation of the initial solution with the presented algorithm for a population size of $pg = 100$ and $gen = 100$ is 10,000 ($pg \cdot$

gen). The selection of the 100 best individuals pg_{best} and the stochastic evaluation of these individuals with $N = 100$ requires another simulation effort of 10,000 simulation runs ($pg_{best} \cdot N$), so that a total simulation effort of $SA_{IKR} = 20{,}000$ simulations must be performed. With a randomly generated initial solution, 10,000 simulations are required for $pg = 100$ and $N = 100$. With $gen = 2$ it´s a total of SA_{rand} =20,000 simulation runs ($pg \cdot N \cdot gen$), so that $SA_{IKR} = SA_{rand}$. The optimisation potential is determined by the relative pairwise comparison between the objective function values of the randomly generated solution ($ICR = 0.0$) and the solutions generated with the ICR concept ($ICR = 0.2 - 1.0$). If, for example, a $T\_M\_PD_{rand}$ value of 100 time units (TU) is obtained with a randomly generated solution and a $T\_M\_PD_{ICR}$ value of 90 TU with a copy rate of 0.2, the optimisation potential is 10 %. A negative optimisation potential means a deterioration when using the ICR concept compared to the random generation of solutions. The results show in general a better performance of the population generated with ICR compared to the population generated randomly (Fig. 3).

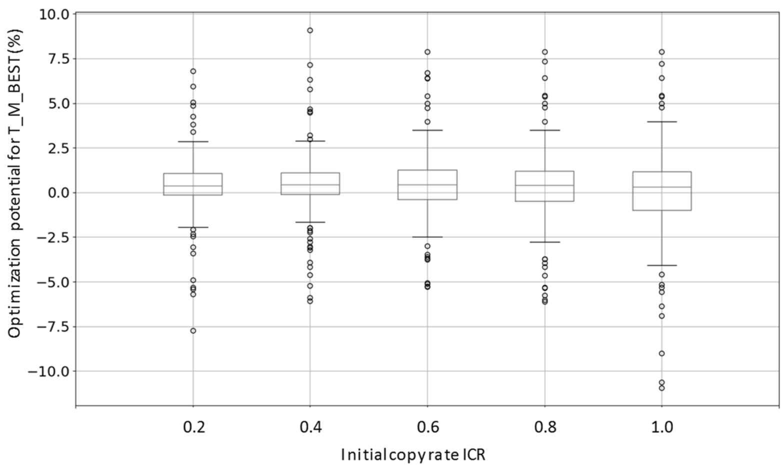

Figure 3: *Boxplot of the fitness optimisation potential comparing ICR=0.2-1.0 to ICR=0.0 (a positive value means improvement of ICR=0.2-1.0 compared to ICR=0.0, a negative value means worsening).*

The median of the fitness optimisation potential is in average 0.5 % of all ICR. The plot also shows that in about 75% of the cases an improvement in fitness values is achieved independently of the ICR, since all quantiles and outliers are in the positive range. Nevertheless, in some cases (about 25 %), the application of the ICR strategy will worsen the result compared to the random strategy. Further experiments are necessary to test whether the ICR can be selected as a function of various influencing factors. If an ICR is to be determined, an ICR of 0.6 is best because it has the least negative outliers. When comparing the 90 % confidence intervals of the optimisation potential (%), the ICR=0.6 also shows the best values ($ICR = 0.2{:}\,0.28 \pm 0.37; ICR = 0.4{:}\,0.39 \pm 0.41; ICR = 0.6{:}\,0.42 \pm 0.41; ICR = 0.8{:}\,0.26 \pm 0.44; ICR = 1.0{:}\,-0.05 \pm 0.53$).

5.3 Comparison to SPR

In a first step, the developed heuristic was compared with SPR (Fig. 4). The proposed sequence heuristic achieved the best result in approximately 60 % of the cases. Further experiments are necessary to limit the scope of the proposed heuristic.

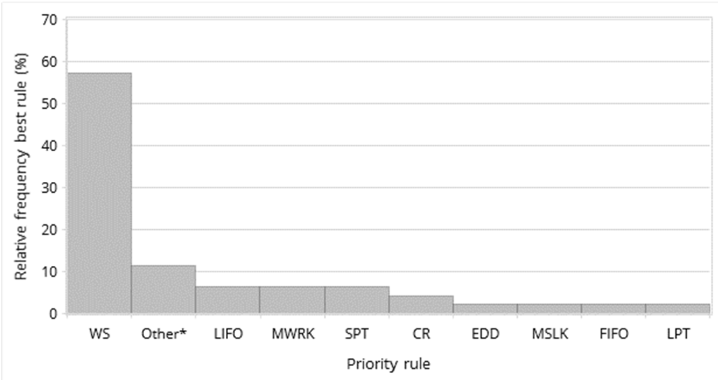

Figure 4: *Performance comparison between proposed weighted sum sequencing heuristic (WS) and common SPR (LIFO: last in first out; MWRK: most work remain; SPT: shortest porcessing time; CR: critical ratio (time remaining until due date devide by remaining processing time); EDD: earliest due date; MSLK: minimum slack (due date-current time-remaining processing time); FIFO: first in first out; LPT: longest processing time).*

6 Conclusions

This paper presents a method for the efficient generation of sequencing heuristics. The efficiency is achieved by selecting good initial solutions as a basis for stochastic optimisation and through deterministic exploration of the solution space. Compared to randomly generated solutions with the same simulation effort, the method achieves better solutions in approximately 75 % of the cases. Deterministic solutions are therefore suitable for stochastic initial solutions. Further investigations are necessary to determine whether it is better to define the initial copy rate as a function of the input parameters. A first comprehensive evaluation of the proposed sequencing heuristic with SPR was promising.

References

Baek, D.H.; Yoon, W.C.: Co-evolutionary genetic algorithm for multi-machine scheduling. International Journal of Production Research 40 (2002) 1, pp. 239–254.

Branke, J.; Hildebrandt, T.; Scholz-Reiter, B.: Hyper-heuristic evolution of dispatching rules. Evolutionary Computation 23 (2015) 2, pp. 249–277.

Branke, J.; Nguyen, S.; Pickardt, C.W.; Zhang, M.: Automated design of production scheduling heuristics. IEEE Transactions on Evolutionary Computation 20 (2016) 1, pp. 110–124.

Frazzon, E.M.; Kück, M.; Freitag, M.: Data-driven production control for complex and dynamic manufacturing systems. CIRP Annals 67 (2018) 1, pp. 515–518.

Freitag, M.; Hildebrandt, T.: Automatic design of scheduling rules for complex manufacturing systems by multi-objective simulation-based optimization. CIRP Annals 65 (2016) 1, pp. 433–436.

Herroelen, W.; Leus, R.: Project scheduling under uncertainty. European Journal of Operational Research 165 (2005) 2, pp. 289–306.

Hildebrandt, T.; Heger, J.; Scholz-Reiter, B.: Towards improved dispatching rules for complex shop floor scenarios. In: Pelikan, M. (Ed.): Proceedings of the 12th annual conference on Genetic and evolutionary computation, Portland, Oregon, USA, 7/7/2010 - 11/7/2010, 2010, pp. 257-264.

Homberger, J.: A multi-agent system for the decentralized resource-constrained multi-project scheduling problem. International Transactions in Operational Research 14 (2007) 6, pp. 565–589.

Homberger, J., 2019: MPSPLIB. Hochschule für Technik Stuttgart. www.mpsplib.com, 15.02.2019.

Jorapur, V.S.; Puranik, V.S.; Deshpande, A.S.; Sharma, M.: A promising initial population based genetic algorithm for job shop scheduling problem. Journal of Software Engineering and Applications 09 (2016) 05, pp. 208–214.

Klemmt, A.: Ablaufplanung in der Halbleiter- und Elektronikproduktion. Dresden, Germany: Springer Vieweg 2012.

Kühn, M.; Schmidt, T.; Genßler, P., 2019: PyScOp: TU Dresden, Professur für Technische Logistik. https://tlscm.mw.tu-dresden.de/scm/git/PyScOp, 15.07.2019.

Luke, S.; Panait, L.: A comparison of bloat control methods for genetic programming. Evolutionary Computation 14 (2006) 3, pp. 309–344.

Omar, M.; Baharum, A.; Hasan, Y.A.: A job-shop scheduling problem (JSSP) using genetic algorithm(GA). Yahya, 2006.

Pickardt, C.W.; Hildebrandt, T.; Branke, J.; Heger, J.; Scholz-Reiter, B.: Evolutionary generation of dispatching rule sets for complex dynamic scheduling problems. Int. Journal of Production Economics 145 (2013) 1, pp. 67–77.

Schlick, J.; Stephan, P.; Loskyll, M.; Lappe, D.: Industrie 4.0 in der praktischen Anwendung. In: Vogel-Heuser, B.; Bauernhansl, T.; ten Hompel, M. (Eds.): Handbuch Industrie 4.0. Vol. 2. Automatisierung, Berlin, Germany: Springer Vieweg 2017, p. 3-29.

Schmidt, T.; Kühn, M.; Genßler, P.: Design of project-oriented calculation models for job priorities by using a customized genetic algorithm. In: Wenzel, S.; Peter, T. (eds.): Simulation in Production and Logistic. Kassel: ASIM 2017, pp. 99-109

Werner, F.: Genetic Algorithms for shop scheduling problems: a survey genetic. In Siarry, P. (Ed.): Heuristics: Theory and Applications, 2013, p. 161 - 222.

Wong, P.; Zhang, M.: Algebraic simplification of GP programs during evolution. In: Cattolico, M. (Ed.): Proceedings of the 8th annual conference on Genetic and evolutionary computation, Seattle, Washington, USA, 8/7/2006 - 12/7/2006, 2006, pp. 927.