Proceedings of the 14th International Symposium on Process Systems Engineering – PSE 2021+
June 19-23, 2022, Kyoto, Japan
http://dx.doi.org/10.1016/B978-0-323-85159-6.50089-0

Evolutionary Algorithm-based Optimal Batch Production Scheduling

Christian Klanke[a*], Engelbert Pasieka[a*], Dominik Bleidorn[b], Christian Koslwoski[b], Christian Sonntag[b], Sebastian Engell[a]

[a]Process Dynamics and Operations Group, TU Dortmund University, Emil-Figge-Straße 70, 44227 Dortmund, Germany
[b]INOSIM Software GmbH, Joseph-von-Fraunhofer-Str. 20, 44227 Dortmund, Germany
{christian.klanke, engelbert.pasieka}@tu-dortmund.de

Abstract

In this work, a simulation-optimization strategy is applied to a benchmark scheduling problem from the pharmaceutical industry, as published by Kopanos, et al. (2010). The optimization is performed by a meta-heuristic using a commercial Discrete Event Simulation software as the schedule builder (simulation-optimization approach). Our work is motivated by commonly encountered real world scenarios where detailed simulation models of the production processes are available and can be used to validate and evaluate the schedules in the presence of many, often non-standard constraints. Moreover, the effort for re-modelling and for the maintenance is reduced by using the available simulator. The meta-heuristic applied here is an Evolutionary Algorithm and we discuss different variants of the encoding of the problem. It is demonstrated that for regular objectives the performance is similar to the tailored MILP-based solution strategy of Kopanos et al. (2010) where a two-stage decomposition strategy is employed.

Keywords: Batch Production Scheduling, Pharmaceutical Industry, Simulation-Optimization, Meta-heuristics, Evolutionary Algorithm, Discrete Event Simulation.

1. Introduction

Simulation-optimization (SO) is a versatile tool for the solution of planning and scheduling problems. In industrial practice, often simulation models of different degrees of accuracy are available from the plant design stage and/or used as a tool in operations for example to validate delivery promises and to determine bottlenecks. Some tools can represent complex constraints of the execution of the orders, include maintenance, the availability of personnel, feedstock and packing materials and the like. Also stochastic effects as e.g. disturbances or varying processing times can often be included. Usually such tools are implemented as Discrete Event Simulators (DES) where rules for the execution of the production can be implemented flexibly. Such simulators enable the end-user to model the production processes in detail and therefore to validate production schedules with respect to constraints which are difficult to formulate otherwise. The models that are built for commercial DES software are typically maintained by the industrial end users, which provides flexibility with respect to changing rules, constraints, recipes or even the set-up of the plant in an intuitive manner. If such a simulation is available for a given plant or process, it is often desired by the user to use it also for planning and scheduling purposes beyond manual generation of plans or schedules and

testing them in simulations. As the models do not conform to a specific mathematical formalism, combining simulation by the DES for the execution and performance evaluation of the schedules with meta-heuristics for schedule generation is an obvious approach. Clearly, this does not provide provably optimal or near-optimal solutions.
In contrast, mathematical programming (MP) provides exact solutions with performance guarantees. It can be applied even to large-scale problems (Harjunkoski, et al., 2014) but often the problem size and complexity leads to unacceptably long computation times or large optimality gaps. This issue is usually dealt with in a semi-heuristic manner, i.e. by employing decomposition approaches that exploit the problem structure in a tailored manner, generating sub-problems with reduced numbers of degrees of freedom. These sub-problems can be solved faster and their solutions are then combined to yield the solution of the full problem (Klanke, et al., 2021, Georgiadis, et al., 2019). MP yields solutions with a measure of optimality and the problems can be solved deterministically to proven optimality. However, as soon as decomposition approaches are used, a measure of optimality usually also is not provided, as only the optimality gaps of the sub-problems are known, but not the optimality gap of the final solution. In addition, the quality of the solutions depends on the heuristics that are employed to perform the decomposition, e.g., the assignments of orders to sub-problems in case of order-based decomposition. A major disadvantage of MP solutions is the need for expert knowledge to formulate the problem at hand and to maintain the models which is far less intuitive than parameterizing a DES.
In related work, a SO approach based on the same commercial simulator (Klanke, et al., 2021) was applied to a complex industrial make-and-pack scheduling problem for which no solutions from exact optimization approaches were available. In this work, we address the well-studied problem from Kopanos, et al., (2010), a large-scale benchmark batch scheduling problem from the pharmaceutical industry, to investigate the quality of the solutions obtained with the SO approach in comparison to those obtained from the tailored MP formulation in Kopanos, et al., (2010). There a two-step MILP decomposition strategy was proposed and the authors stated that "[…] a comparison of the solution method with elaborated metaheuristics would be of great interest." The remainder of this contribution is structured as follows: We start by giving a short overview of the case study and its key features in Section 2. Then, in Section 3, we introduce our methodology, including the representation of solutions and the genetic operators. In Section 4, we present the results of our approach for regular objectives and compare them with those obtained by Kopanos, et al. (2010). Additionally, we present and discuss results with our proposed approach for the non-regular Weighted Lateness objective where timing decisions had to be added to our approach. In the last Section we conclude our findings and present an outlook.

2. Case Study

The case study addressed in this work is taken from Kopanos, et al., (2010). It is a multiproduct batch plant with 17 units (machines) that are organized in 6 stages. The problem, which is a variant of a hybrid flow shop problem, comprises 12 instances that vary in the number of orders, the objectives, and in the storage policy. The features of the problem include limited product-unit flexibility, machine-dependent processing times, sequence-dependent changeover times, and product-specific recipes, meaning that certain jobs are not processed on some of the available stages.

This paper considers three problem instances with 30 orders, unlimited intermediate storage (UIS), and the objectives Makespan (C_{max}), Overall & Changeover Cost (O.&C.C.) and Weighted Lateness (W.L.), which are minimized. The objectives are defined as

$$C_{max} = \max\left(C_1, .., C_{|I|}\right)\ [h] \tag{1}$$

$$O.\&C.C\ = \omega C_{max} + \Sigma_{i\in I}\ cc_i\ [h], \text{with } \omega = 0.9 \cdot 10^3 \tag{2}$$

$$W.L. = \Sigma_{i\in I}\ \alpha E_i + \beta T_i\ [h], \text{with } \alpha\ = 0.9 \text{ and } \beta\ \ 4.5 \tag{3}$$

C_i denotes the completion time of job i, cc_i denotes the sum of all changeover times multiplied with a sequence dependent impact factor associated with job i, and E_i and T_i denote the earliness and the tardiness of job i.

3. Methodology

In this section, a generic solution approach that works without decomposition of the problem is presented. We first focus on the regular objectives C_{max} and O.&C.C..
Our solution method uses modular representations for the different degrees of freedom that can be adapted according to the problem at hand. The case study has three generic degrees of freedom: the allocation of jobs or rather of their operations to units, the sequences of operations on units, and the timing of the operations. Our approach uses separate strings for all decisions similar to the approach presented in Chen, et al., (1999), to maintain flexibility in the choice of the representation. To reduce the search space, heuristics can replace some of these decisions. We tested different combinations of representations for the sequencing and the allocation. Encoding timing decisions explicitly was not necessary for the regular objectives, C_{max} and O.&C.C., because no improvements of the solution quality can be obtained if delays in the starting times of operations are included.

3.1. Encodings

For the two objectives C_{max} and O.&C.C., an encoding of the sequences and, depending on the applied strategy, an encoding of the allocation to units is used.
To keep the dimension of the search space manageable, the global sequence of orders, i.e. a single sequence $\pi\ =\ (\pi_1, \dots, \pi_{|I|})$ permuting the set of jobs $i \in I$, which is imposed on all stages, is employed. In the simulation, this sequence is decoded by the DES software by processing the jobs in the order of appearance in the global sequence π. Consequently, the sequences of operations on each stage are tightly coupled, i.e., if job i follows i' on unit M01, i' cannot finish before i if they follow the same recipe and are therefore processed on the same units. However, as shown in (Kopanos, et al., 2010), in the schedule for the 30-product case, minimizing C_{max} under UIS-policy, better solutions can be obtained when the sequence of operations on the same units are swapped within two consecutive stages. However, encoding individual sequences for all stages S, would increase the search space significantly and lead to the need of a much larger number of calls of the simulator by the EA.
Two different ways to decide on the allocation of jobs to units were investigated. One option is to determine the allocation dynamically during the simulation using a rule that is implemented in the simulator such that the highest-priority operation is allocated to and executed by the machine that first becomes idle, and started as soon as it becomes idle,

leading to non-delay schedules. So the unit on which an operation is processed in each stage is determined by an EST heuristic.
In the second option, the allocation encoding assigns to every operation $o_i \in O_i$ of a job i a unit $u_s \in U_s$ in the corresponding stage $s \in S$. The allocation encoding $\alpha \subset O \times U$ therefore is a partial relation of the set of all operations $O = \bigcup_{i \in I} O_i$ and the set of all machines in all stages $U \in \bigcup_{s \in S} U_s$.
Employing these encodings two optimization strategies are obtained: *Strat1*, where the global sequence and the allocation are optimized simultaneously and *Strat2*, where the global sequence is optimized and the EST heuristic is used for the determination of the allocation during the simulation.
Weighted lateness is a non-regular objective where timing decisions are important, because the objective value is non-decreasing with the completion time of the scheduled jobs (Baker and Scudder, 1990). For the W.L. minimization, a simple heuristic improvement strategy was used for pre- and post-processing of the EA solutions. Prior to the optimization, the jobs were sorted according to their earliest due dates. The number of possible job permutations is still very high, because many jobs share the same due date. In a simple repair step, each operation on the last stage was delayed if it was early and the delay would not increase the tardiness of a following job.

3.2. Genetic operators

The parent and survivor selection operators are identical to those that were used in Klanke, et al. (2021), i.e. a rank-based parent selection and a rank-based/elitist survivor selection. In the latter operator, a fixed percentage of the best individuals are guaranteed to survive, while all remaining individuals are chosen via rank-based selection. Parameters for which the values are not stated explicitly in this work are also chosen as in Klanke, et al. (2021).
For the allocation chromosome, the Point Mutation operator that randomly picks an operation o and assigns it to a new unit of the same stage, and a Uniform Crossover operator that iterates over all products and stages and assigns a new unit, either from parent P1 or parent P2, with equal probability, is used.
As the mutation operator for the global sequence chromosome, we employ the Permutation Mutation operator from Eiben and Smith (2015) that cuts a sub-sequence of random length and permutes its elements before reinsertion into the chromosome. As the crossover operator, the Cycle Crossover, as reported in Larranga, et al. (1996) is used.

4. Results

In this section the solutions obtained by *Strat1* and *Strat2* are presented. The optimization was run on an i7-7700K Intel CPU under Windows 10 for approx. 2.5 h of computational time. As schedule builder, the commercial software INOSIM 13.0 which, on average, took about 2.5 seconds for a single fitness evaluation, is used. Within a computational time budget of 2.5 h, by parallelization of the fitness evaluation of all individuals of the same generation, in total 3040 evaluations could be performed per problem instance. The results of our approach are presented in Table 1 together with the results from Kopanos, et al. (2010), where the computation time was limited to 1 h. This led in some instances to non-feasible solutions for the monolithic approach (see O.&C.C. in Table 1 in Kopanos, et al. (2010)). The solutions found by the two-step MILP decomposition approach in Kopanos, et al. (2010), the construction (MILP decomp. 1st Step) and the improvement (MILP decomp. 2nd Step), serve as a benchmark for the strategies proposed in this paper. The runs for *Strat2* were repeated several times to evaluate the reproducibility of the solution. *Strat1* could barely reach the solution quality of the MILP monolithic approach for all three objectives, whereas the second strategy Strat2 outperformed the 1st step solution in Kopanos, et al. (2010), and leads to a solution quality between the 1st and 2nd step solutions of the MILP decomposition approach. The best solution that was observed with *Strat2* for the Makespan objective came very close to the 2nd step MILP solution. For the O.&C.C. objective, the best solution was slightly better than the 1st step solution from the decomposition approach. For the W.L. the best run led to a value of 47.22 h, which was reduced to 37.07 h after the repair step. Clearly here a tailored improvement strategy is needed.

Table 1: Results of the EA and MILP approaches for the 30 batches case from Kopanos, et al. (2010). $\mu^1 = 5, \lambda^2 = 40$ and $N_{gen}{}^3 = 600$

Solution Approach	Makespan [h]	O.&C.C. [h]	W.L. [h]
MILP monolithic	34.81	-	428.15
MILP decomp. 1st Step	28.51	66.16	48.16
MILP decomp. 2nd Step	26.56	62.91	19.09
Strat1 (Alloc. + Seq. Enc.)	35.02	77.49	693.71
Strat2 (EST + Seq. Enc.)[4]	27.90 ± 1.17	66.17 ± 0.57	47.78±0.51
Best result from Strat2	26.72	65.59	47.22
After repair step	-	-	37.07

[1]Number of children, [2]Population size, [3]Number of generations, [4]Mean and standard deviation of three runs

5. Summary, Conclusion and Outlook

In this work, we investigated the potential of a simulation-optimization approach, combining an EA and DES, for a benchmark scheduling problem, from the pharmaceutical industry.

Our proposed approach benefits from the use of existing models and only encodes the essential degrees of freedom, while the detailed schedules are built by the simulation system. This has the advantage that all constraints that are implemented in the simulator are respected by the solution so the resulting schedule is executable to the best of the available knowledge of the processes in the plant.

For the case study under consideration, the allocation and the sequence degree of freedom were encoded explicitly (*Strat1*), or only the global sequence of jobs was encoded and the allocation was determined heuristically by the simulator (*Strat2*). The encoding of only the global sequence together with the heuristic allocation provided better results due to the smaller search space of the EA. For the three investigated objective functions, the best results are between the 1st and 2nd step solutions of the benchmark approach. From a practical point of view, the solution quality can be considered as sufficient and the small differences are outweighed by the advantages of the simulation-based approach of intuitive modelling, re-use of models and the ability to implement and modify all kinds of constraints in the execution of the schedule. For the W.L. objective, the optimization of a global sequence of the orders turned out to be insufficient. Here a tailored second-stage solution is needed where the allocation and sequencing decisions on the stages are considered explicitly.

The computation time of the detailed simulation models of a commercial simulation environment is significantly higher than that needed for computing a solution with a simple job-shop model due to the larger overhead that is caused by the possibility to implement more detailed models. It can be reduced significantly by a parallelization of

the fitness evaluation. In our case, in a time span of 2.5 h the EA generated results that are similar to those obtained by the MILP decomposition approach for the Makespan objective.
Overall, the combination of a detailed discrete-event simulation and an evolutionary algorithm is attractive from an industrial point of view because of the flexibility to implement non-standard features in the simulation model, the fact that the modelling is more intuitive and the model can be modified and maintained by plant personnel. For timing-related objectives and large problem sizes, further work on suitable refinement strategies is needed.

Acknowledgements

This work was partially funded by the European Regional Development Fund (ERDF) in the context of the project OptiProd.NRW (https://www.optiprod.nrw/en).

References

Kenneth R. Baker and Gary D. Scudder, 1990, Sequencing with earliness and tardiness Penalties. A review, Operations Research, 38(1), 22–36

A.E. Eiben and J.E. Smith, 2015, Introduction to Evolutionary Computing, Springer-Verlag Berlin Heidelberg

Iiro Harjunkoski, Christos T. Maravelias, Peter Bongers, Pedro M. Castro, Sebastian Engell, Ignacio E. Grossmann, John Hooker, Carlos M´endez, Guido Sand, and John Wassick, 2014, Scope for industrial applications of production scheduling models and solution methods, Computers and Chemical Engineering, 62, 161–193

Christian Klanke, Dominik Bleidorn, Christian Koslowski, Christian Sonntag, and Sebastian Engell, 2021, Simulation-based Scheduling of a Large-scale Industrial Formulation Plant Using a Heuristics-assisted Genetic Algorithm, In GECCO '21: Proceedings of the Genetic and Evolutionary Computation Conference Companion, Association for Computing Machinery, 1587–1595

Christian Klanke, Vassilios Yfantis, Francesc Corominas, Sebastian Engell, 2021, Short-term scheduling of make-and-pack processes in the consumer goods industry using discrete-time and precedence-based MILP models, Computers and Chemical Engineering, 154, 107453

Georgios P. Georgiadis, Apostolos P. Elekidis, Michael C. Georgiadis, 2019, Optimization-Based scheduling for the Process Industries: From Theory to Real-Life Industrial Applications, 7(7), 438

Georgios M. Kopanos, Carlos A. Méndez, and Luis Puigjaner, 2010, MIP-based decomposition strategies for large-scale scheduling problems in multiproduct multistage batch plants: A benchmark scheduling problem of the pharmaceutical industry, European Journal of Operational Research, 207(2), 644–655

Pedro Larranaga, Cindy M. H. Kuijpers, Roberto H. Murga, and Yosu Yurra-Mendi, 1996, Learning Bayesian Network Structures by Searching for the Best Ordering with Genetic Algorithms, IEEE Transactions on Systems, Man, and, Cybernetics-Pakt A: System and Humans, 26(4), 487–493

Haoxun Chen, Jürgen Ihlow, Carsten Lehmann, 1999, A Genetic Algorithm for Flexible Job-Shop Scheduling, 1999 International Conference on Robotics & Automation, 1120-1125