A GENERAL VARIABLE NEIGHBORHOOD SEARCH FOR SIMULATION-BASED ENERGY-AWARE FLOW SHOP SCHEDULING

Bernhard Heinzl
Wolfgang Kastner

Institute of Computer Engineering
Automation Systems Group
TU Wien
Treitlstraße 1-3
A-1040 Vienna, Austria
{bernhard.heinzl,wolfgang.kastner}@tuwien.ac.at

ABSTRACT

Simulation-based optimization methods can support decision-making in industrial Production Planning and Control (PPC) to include energy considerations as part of a multi-objective scheduling problem. In this paper, we investigate a General Variable Neighborhood Search (GVNS) metaheuristic for simulation-based energy-aware production scheduling, including optimization of setup times for thermal processes. The fitness evaluation is based on a hybrid discrete/continuous simulation model that provides assessment of energy costs including time-dependent energy pricing together with other production goals, like storage costs, while also considering dynamic interdependencies between production machinery and building facilities. We demonstrate the method on a case study of flow shop scheduling in an industrial bakery in different scenarios. Comparisons show a potential gain of 6% when considering energy costs as an optimization target within PPC.

Keywords: GVNS, hybrid simulation, hyPDEVS, simulation-based optimization, production planning

1 INTRODUCTION

Energy efficiency in industrial production has become an important topic in recent years due to legislative pressure, but also because of the substantial potential for energy savings in the industrial sector (Chan, Kantamaneni, and Allington 2015). Energy-aware Production Planning and Control (PPC) strategies can be used to influence energy costs during operation. For example, the production of energy-intensive products can be shifted to the night hours where energy is often cheaper, or to the midday hours in case more solar energy is available. However, it is not sufficient to only consider energy as an optimization goal. Instead, energy efficiency must be seen as part of a multi-objective system of production targets together with production variables such as storage costs, throughput times or delivery delays. Such multi-objective problems with complex, sometimes time-dependent constraints are hard to solve for real-world problems. Modern solutions often rely on heuristic or metaheuristic methods.

According to (Wari and Zhu 2016), metaheuristics and customized multi-objective heuristic approaches are well-suited for applications in real-life industrial production planning problems (which typically are NP-hard), in contrast to exact approaches that usually require simplified models. Metaheuristics allow to explore the search space more efficiently and effectively, especially if they are tailored to the individual problem (Güller, Uygun, and Noche 2015). Among these, Variable Neighborhood Search (VNS) algorithms have shown excellent capability for solving scheduling problems (Roshanaei, Naderi, Jolai, and Khalili 2009). This is in accordance with other publications, e.g. (Yazdani, Amiri, and Zandieh 2010, Roshanaei, Naderi, Jolai, and Khalili 2009), which have successfully applied VNS for job scheduling problems in the production domain. In (Gansterer, Almeder, and Hartl 2014), the authors compare different optimization methods for simulation-based optimization of production plans, in which VNS also leads to the best results.

For evaluating the fitness of solution candidates during metaheuristic search, simulation-based methods are gaining interest because they enable to capture complexity of real-world problems including difficult dynamic interactions without the limiting assumptions many other approaches have. However, with regard to energy optimization, interdisciplinary holistic simulation models are required that include dynamic interactions across engineering domains in order to get an accurate prediction of the overall energy demand, that not only includes production machinery, but also technical building services. For example, heating a production oven generates waste heat that is dissipated into the room and affects heating and cooling energy demand for the building. Similarly, the actual setup time for preheating the oven depends on different conditions, including which products have been produced before, and the setup time affects production throughput and scheduling. Incorporating energy considerations in production logistics simulations with their time-dependent interactions in an accurate manner requires advanced modeling and simulation approaches that combine discrete (product flow) and continuous dynamics (energy flow). Describing such hybrid discrete/continuous simulation models in a consistent manner with formally sound semantics in complex real-world applications is still challenging (Brailsford, Eldabi, Kunc, Mustafee, and Osorio 2019).

In our work, we aim to support modern PPC tools in incorporating energy considerations into the planning process using simulation-based optimization techniques. In this paper, we present a metaheuristic procedure for energy-aware optimization of production scheduling and demonstrate it on a case study. The procedure employs a dynamic discrete/continuous simulation model for evaluating overall energy demand as well as material flow, while taking into account dynamic interdependencies across domains. To this end, we have developed a component-based hybrid simulation tool based on a formal model description, called hyPDEVS, which is described in (Heinzl and Kastner 2019). The simulation tool includes reusable model components that can be used to simulate interdisciplinary production systems. The optimization method aims at sequencing and time scheduling a given list of production jobs while minimizing energy demand together with other production goals. This includes optimizing setup times to reduce energy demand and increase production throughput. The procedure combines a Variable Neighborhood Search (VNS) metaheuristic with Variable Neighborhood Descent and Simulated Annealing (SA) for local search and diversification. We demonstrate the feasibility of this method on a flow shop scheduling problem of an industrial bakery. The simulation model includes a production line with alternative paths as well as technical building services for energy supply and a thermal building model. We compare different scenarios and highlight the potential benefit of considering energy as an optimization target.

2 CASE STUDY OVERVIEW

To evaluate the described method on a real-life example, we devised a simplified model of a real production plant of an industrial bakery that produces baked goods (Heinzl and Kastner 2019, Heinzl 2020). The model features a typical production line with machines, storage and conveyor belts, an energy supply system with heater and cooler, and a building model with thermal zones.

The *production and logistics* components form a production line for two product variants: baked and frozen. Baked products pass a oven for baking while frozen products are frozen directly (in a freezer) without being baked. Both are designed as conveyor belts, meaning that new entities continuously enter the stations and

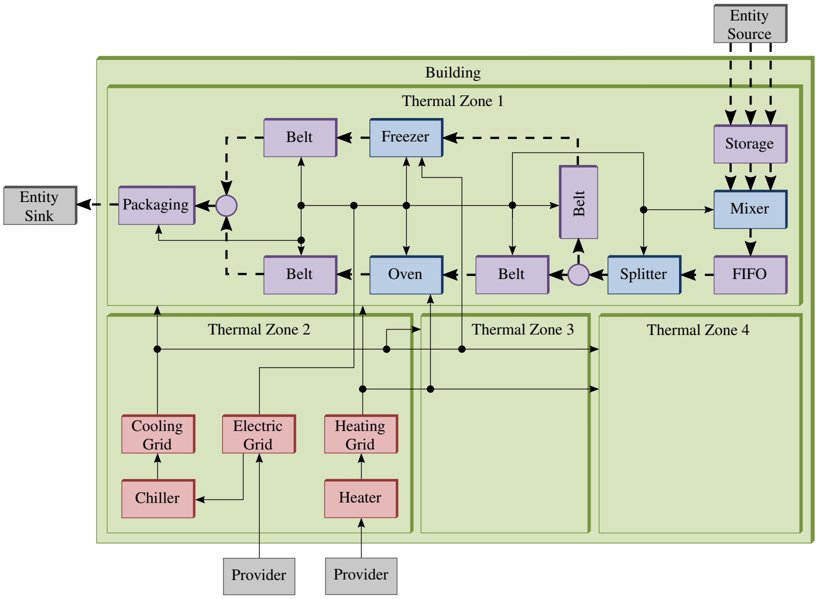

Figure 1: Case study production facility, consisting of production machines (blue), logistics components (purple), technical building services for energy supply (red) and thermal building zones (green)

leave on the other side. The *building* is modeled as a simple thermal compartment model with four thermal zones, each representing a distinct part of the facility: production hall, plant rooms and cold storage. These zones all have independent conditioning (for example, the cold storage is kept at 4 °C) and exchange thermal energy with one another according to the defined wall topology. They also exchange thermal energy with the environment, for which a variable ambient temperature may be specified. The *energy system* provides necessary technical building services, mainly supplying energy for the production machines as well as for heating and cooling the thermal building zones.

3 SIMULATION-BASED OPTIMIZATION STRATEGY

Production optimization deals with finding an optimal combination of production resources, such as equipment, utilities or energy, to achieve a given production target in the best possible way. Operative Production Planning and Control (PPC) methods consider production resources and cost factors to find optimal production schedules. To support PPC in practice, modern APS (Advanced Planning and Scheduling) software systems offer integrated resource planning (Kilger, Meyr, and Stadtler 2015, Heinzl 2020). The complexity of the underlying optimization problem with multiple competing objectives and complex constraint conditions suggests using simulation methods as part of the objective function (Swisher, Hyden, Jacobson, and Schruben 2000). Hereby, a simulation model of the system under consideration serves as a prescriptive tool to predict and evaluate the performance of a given scenario. Dynamic simulation allows to consider more complex systems than with conventional analytical models, especially for highly time-dependent problems, while offering more accurate predictions and overall improving planning quality.

However, the fact that these methods in general do not provide closed analytical representations of the objective function (or its derivatives) prevents straightforward deployment of many standard optimization algorithms. Instead, many practical simulation-based optimization solutions employ metaheuristics that rely solely on the evaluation of the objective function itself. These algorithms modify a candidate solution in an iterative manner to find a near-optimal solution until termination criteria are met. The final solution may not be the global optimum, but is often good enough in practice.

The overall computation cycle of using the simulation-based optimization methodology for production scheduling in industry is illustrated in Figure 2. The starting point is a given demand plan (Dplan) specifying how many of which entities (i.e. products) need to be delivered when. This Dplan serves as a basis to generate an initial solution of a production schedule (Pplan), which is then evaluated using a dynamic simulation model. The results of the simulation are fed back to the optimization to be evaluated for its fitness using a specified cost function $f(x)$ (objective function). Based on this evaluation, the optimization algorithm iteratively adapts the solution in order to find the minimum of $f(x)$ subject to constraints. The cycle continues until certain termination criteria (e.g. fixed number of iterations, computation time threshold, etc.) are met.

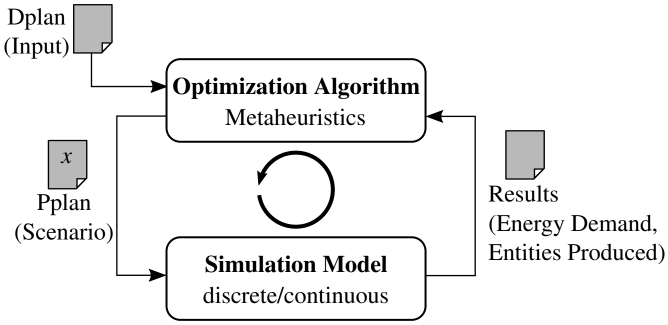

Figure 2: Simulation-based optimization cycle. Input is a demand plan (Dplan) of entities to be produced. The optimization algorithm generates a production schedule (Pplan), which is adapted iteratively based on an evaluation of simulation results.

4 METAHEURISTIC OPTIMIZATION METHOD

We employ a single-solution metheuristic method based on a General Variable Neighborhood Search (VNS) procedure. The VNS (Roshanaei, Naderi, Jolai, and Khalili 2009) is an effective method guiding a local search by switching between increasingly larger neighborhoods to efficiently explore the solution space. The VNS consists of (1) a shaking phase that generates randomized perturbations of the solution for diversifying the search to escape local optima, and (2) an intensification phase that searches for improvements in the local neighborhood, typically using a local search procedure. For improving the performance of the search, the Generalized VNS (GVNS) replaces the local search by a Variable Neighborhood Descent (VND) procedure (Siarry 2016, Hansen and Mladenović 2009). The VND is similar to the VNS, but does not include shaking and thus limits the search to strict improvements (i.e. descent). It does not use the same neighborhood structures as the VNS, which offers more flexibility in tailoring the search procedure to the problem instance.

Algorithm 1 presents an overview of the optimization method as pseudocode. After generating a feasible initial solution x, the VNS component (described in more detail in Section 4.3) guides a search for improvements over a fixed number of iterations n_{iter} by generating in each iteration a random neighboring solution x' (by means of the neighborhood structure $\mathcal{N}_k(x)$) that serves as starting point for the VND that tries to find a local minimum x''. If the solution is accepted, the VNS restarts with neighborhood $\mathcal{N}_1$ and the new starting solution x''. Otherwise, the neighborhood is enlarged. The acceptance criterion is based on Simulated Annealing (SA) (Siarry 2016) for further diversifying the search by allowing to accept potentially worse solutions during the early search phase. Infeasible solutions are also allowed during the search and are evaluated based on a penalizing cost function, which is described in Section 4.2. For the particular case study, the optimization vector x contains the job starting times and the setup time duration for oven and freezer, which allows to optimize job sequencing and scheduling as well as setup processes, which also have an influence on the energy demand.

Algorithm 1 GVNS/SA algorithm

```
$\mathcal{N}_k \leftarrow$ set of VNS neighborhood structures for $k = 1, \ldots, k_{\max}$
$x \leftarrow InitialSolution()$
$k \leftarrow 1;\ i \leftarrow 0;\ T \leftarrow T_0$
while $i \leq n_{\text{iter}}$ do
    $x' \leftarrow shaking(\mathcal{N}_k(x))$        ▷ generate random perturbation
    $x'' \leftarrow VND(x', \mathcal{N}_k)$           ▷ local improvement search
    if $acceptSA(x'', x, T)$ then           ▷ acceptance based on SA
        $x \leftarrow x''$
        $k \leftarrow 1$
    else
        $k \leftarrow (k \bmod k_{\max}) + 1$         ▷ try next neighborhood
    end if
    $i \leftarrow i + 1$
    update $T$
end while
```

4.1 Initial Solution

An initial solution (function *InitialSolution*()) is generated by employing a custom construction heuristic, which takes the jobs from the provided input list (Dplan), sorts them in increasing order of due date and then schedules the jobs in this order as soon as possible (forward scheduling) while considering safety gaps between jobs as well as setup times. This method ensures that no collisions occur, however, further constraint violations, such as delivery delay, or energy considerations, are not checked during this phase.

4.2 Generalized Cost Function

The algorithm allows infeasible solution during the search process. Constraints are penalized by means of the generalized cost function (Heinzl 2020)

$$f(x) = \omega_1 \cdot f_{\text{dev}}(x) + \omega_2 \cdot f_{\text{en}}(x) + \omega_3 \cdot f_{\text{st}}(x) + \omega_4 \cdot f_{\text{del}}(x) + \omega_5 \cdot f_{\text{sep}}(x) \tag{1}$$

where f_{dev} denotes a penalization for deviating number of entities produced, f_{en} the energy costs, f_{st} the storage costs, f_{del} is a penalization for any potential delivery delay, and f_{sep} penalizes job separation violations. These part goals are evaluated based on the simulation. The energy costs f_{en} are calculated by taking the simulated power supplied by the energy providers (cf. Figure 1), rate them with time-dependent energy prices and accumulate them over time to obtain the overall energy costs. This approach of using variable energy prices allows to take into account different effects that potentially influence energy-aware production planning, like lower energy prices during nighttime or due to an additional photovoltaic system that provides solar energy during the day. The storage costs f_{st} are determined based on the time difference between job completion and delivery due date (given in the demand plan). For more details, we also refer to (Kamhuber, Sobottka, Heinzl, and Sihn 2019). The coefficients $\omega_i > 0$ are weighting factors, which may be adapted by the user to balance their preferences for individual part goals and trade-offs in a transparent manner. This *weighted sum* method is common for multi-objective optimization problems in practice as it is easy to implement and intuitive for the user. However, the results are often highly dependent on the weights (Freitas 2004).

4.3 Neighborhood Structures

The general operation of the VNS has been described in Algorithm 1. It is responsible for diversifying the search in a structured way during the shaking phase by successively changing the neighborhood structures $\mathcal{N}_k$. These neighborhood structures are usually defined implicitly by means of operators that modify the solution. For our case, we have defined four different operators:

1. *OpSwitch:* This operator changes the order of jobs by taking a random number r of successive jobs (starting from a random position) and moving them to a different position. Hereby, r is chosen in the interval $r \in [1, \min\{r_{\max}, n\}]$, where n is the overall number of jobs and $r_{\max}$ changes depending on the neighborhood k, see Table 1.
2. *OpShift:* The shifting operator takes a random position an moves all subsequent jobs by a specified time t_{shift}, where t_{shift} depends on k.
3. *OpChangeSetuptime:* Here, the setuptime is changed by a specified an amount t_{setup}.
4. *OpMerge:* The merging operator takes two random jobs, which are removed by a distance of d, and which have the same product type and combines them into one job. Merging jobs has the advantage that it reduces the number of setup processes and avoids gaps between jobs, thereby increasing production and energy efficiency.

The VND, which is used in place of a local search procedure inside the VNS to improve the generated solution in the intensification phase (Hansen and Mladenović 2009), uses the same basic operations, albeit in different neighborhoods depending on the current VNS neighborhood $\mathcal{N}_k(x)$. For the switching operator, the VND neighborhood containing all pairs of successive jobs is explored exhaustively. The same is done with respect to merging. For *OpShift*, the VND checks shifting groups of jobs in a binary search pattern by a set of different times t_{shift}. Similarly for the setup times, which are being reduced iteratively by a set of different t_{setup}. The complete neighborhood structures used for the case study are presented in Table 1.

Table 1: Neighborhood structures used in the GVNS

k	**Operator**	**Shaking**	**VND**
1	1	$r_{\max} = 2$	$r_{\max} = 1$
2	1	$r_{\max} = 4$	$r_{\max} = 1$
3	2	$t_{\text{shift}} = 8\,\text{h}$	$t_{\text{shift}} \in \{4, 2, -1, 0.5\}\,\text{h}$
4	2	$t_{\text{shift}} = 12\,\text{h}$	$t_{\text{shift}} \in \{4, 2, -1, 0.5\}\,\text{h}$
5	3	$t_{\text{setup}} = 0.5\,\text{h}$	$t_{\text{setup}} \in \{0.5, 0.25\}\,\text{h}$
6	4	$d = 2$	$d = 1$

4.4 Acceptance Criterion

Instead of accepting only improving solutions, a modified acceptance criterion (*acceptSA*()) is used that is based on a Simulated Annealing (SA) method (Siarry 2016). It enables to further diversify the search and better escape local optima by accepting potentially worse solutions, albeit with decreasing probability as the search progresses. To be more precise, while an improving solution is always accepted, a deteriorating solution x'' is accepted with the probability $p_{\text{SA}} = \exp\left(-(f(x'') - f(x))/T\right)$, where f is the generalizes cost function (see Section 4.2) and T is the temperature that decreases linearly after every VNS iteration in such a way that $T < 10^{-3}$ during the last 10% of iterations. This effectively tightens the acceptance criterion as the search progresses until, finally, only improving solutions are accepted at the end. The temperature T is initialized with T_0 according to $T_0 = -\Delta_{\text{SA}}/\log(0.5)$, meaning that, initially, a solution being Δ_{SA} worse than $f(x)$ is accepted with a probability of 50%. Experiments revealed a value of $\Delta_{\mathrm{SA}} = 15$ to be suitable for the case study.

5 HYBRID SIMULATION

As already mentioned, it is important to use hybrid discrete/continuous simulation methods in the context of interdisciplinary assessment of energy efficiency in production. These allow both the material flow to be accurately modeled as Discrete-Event system and the energy flow by means of differential equations, while also taking into account dynamic interactions between these domains. Continuous representation of energy flow, as opposed to discrete energy profiles, enables to accurately incorporate transient dynamics, for example the heat-up process of an oven or the thermal heat capacity of the building. For hybrid modeling and simulation, we employ a formal model description, called hyPDEVS, which is based on the *Discrete Event System Specification* (DEVS) (Zeigler, Prähofer, and Kim 2000). The following provides a brief introduction into the formalism in order to give the necessary background.

5.1 hyPDEVS Formalism

The hyPDEVS formalism is a formal model description based on the *Discrete-Event System Specification* (DEVS) as an extension for incorporating continuous model aspects into a discrete-event model description (Deatcu and Pawletta 2012).

The formalism allows to build models from components in a hierarchical manner by distinguishing between *atomic* and *coupled* components. More formally, a hyPDEVS *atomic* is specified by the tuple

$$M = \left\langle X, Y, S, f, c_{se}, \lambda_c, \delta_{state}, \delta_{ext}, \delta_{int}, \delta_{conf}, \lambda_d, ta \right\rangle, \tag{2}$$

with the sets of input events X, outputs Y and states S, all of which may contain discrete as well as continuous values. The remaining entries in (2) are functions specifying the continuous and discrete behavior: rate of change function f describing ordinary differential equations (ODEs), continuous and discrete output functions λ_c and λ_d, state event condition function c_{se} for localizing state events, and transition functions δ_{state}, δ_{ext}, δ_{int} and δ_{conf} for state event, external, internal and concurrent (confluent) transitions, respectively.

Coupled hyPDEVS models are comprised of an external interface (input/output), sub-components (which must again be hyPDEVS components) and coupling relations. Coupled systems can be arranged hierarchically, meaning they can be incorporated just like an atomic into a larger coupled system. This property allows to construct modular hierarchical (tree-like) models in a component-based manner. Components also facilitate separation of concerns in order to be able to manage the complexity of large-scale simulation models. More details ar given in (Heinzl 2016, Zeigler, Prähofer, and Kim 2000).

5.2 Case Study Implementation

Based on the described hyPDEVS formalism, a simulation tool was implemented in C++ (Heinzl and Kastner 2019) together with a library of model components for modeling production systems, which are also used in the case study, see Figure 1. The production stations, especially oven and freezer, are modeled as hybrid discrete/continuous models, where the discrete model is responsible for entity flow and control logic, and the continuous model handles energy input and conversion. In particular, the energy conversion follows simple energy balance equations, including thermal heat capacity of the station, and generates diffuse waste heat that is dissipated into the respective thermal zone in the room model. The energy model also interacts with the material flow by means of state events that indicate e.g. when the oven has reached its target temperature and is ready to accept entities. More details are described in (Heinzl 2016). Outside energy providers supply the energy across the system boundary to the energy system, which is billed to the customer using time-dependent energy pricing. Specifying energy price profiles allows to incorporate various energy cost effects that influence production planning. To demonstrate the effect of variable energy pricing, we specify reduced price for electric energy between 8 a.m. and 6 p.m. on selected days for this case study, see the plot in Figure 6. The idea is that this accounts for cheaper energy from a photovoltaic system during sunshine.

6 CASE STUDY EXPERIMENTS

This section demonstrates the application of the proposed GVNS/SA method on the flow shop scheduling case study (Heinzl 2020). The optimization was implemented as a prototype in MATLAB and coupled with the standalone hyPDEVS simulator (see Section 5.2). Table 2 presents two typical scenarios of demand lists (Dplan) needing to be scheduled over the course of one week (i.e. 168 h simulation time). The table lists different orders (coming from customers) with quantities and delivery due dates (measured from the start of the week). Duplicate entries constitute different orders to be delivered to different customers.

Table 2: Demand plan for the example scenario

Product Type	**Quantity**	**Due Time**	
		Scenario 1	Scenario 2
baked	6	48 h	168 h
baked	6	48 h	168 h
frozen	20	48 h	168 h
frozen	20	48 h	168 h
baked	24	72 h	168 h
frozen	50	72 h	168 h
baked	36	120 h	168 h
frozen	50	120 h	168 h
baked	8	144 h	168 h
baked	8	144 h	168 h
baked	8	144 h	168 h
frozen	25	144 h	168 h
frozen	25	144 h	168 h
baked	20	168 h	168 h
frozen	75	168 h	168 h

Scenario 1 is based on a real demand plan and features two different products (baked and frozen) in different batch sizes and with different due dates. Figure 3 shows the progression of the cost function over the course of iterations. The simulation time was 1 week, and the total number of VNS iterations was chosen as $n_{\text{iter}} = 100$. Further experiments showed that increasing the number of iterations does not improve the end result significantly anymore. For testing purposes, the weighting factors for the part goals were chosen as $\omega = (\omega_i)_{i=1\ldots5} = (1, 2, 0.5, 1, 1)$. When looking at the part goals, it shows that no significant constraints have been violated that would have caused penalties (entity deviation, delivery delay) and that the majority of the costs is divided between storage costs and energy costs. Overall, the cost function could be reduced by about 34%, with energy costs reduced by 30% and storage costs by almost 50%.

Figure 5 depicts in more detail the oven and freezer allocations (i.e. number of entities and temperature over time) in the initial solution, which was generated using the simple construction heuristic from Section 4.1, against the final optimization results. The different humps in the entity count correspond to different jobs. The results show that the jobs are being grouped together around the due dates and that some are produced earlier than necessary, which saves on storage costs as well as energy costs, on the one hand by avoiding

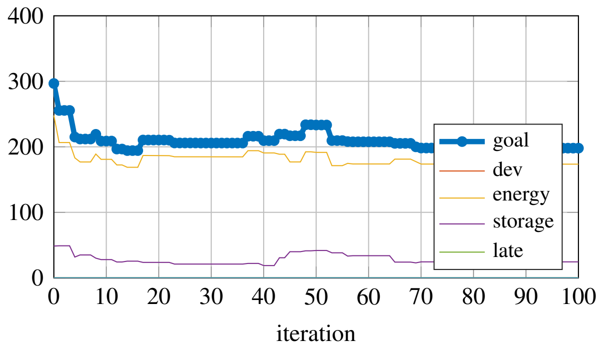

Figure 3: Progression of the cost function and part goals for Scenario 1

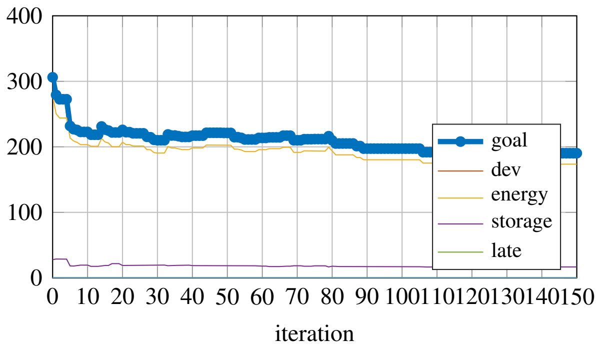

Figure 4: Progression of the cost function and part goals for Scenario 2

unnecessary setup processes (e.g. reheating the oven), on the other hand also by optimizing the operating times. The gaps, where the stations are idle, increase, which allows to switch off the oven and freezer (shown in the figure) instead of continuing them to run, which would consume unnecessary energy.

This effect becomes even more apparent in Scenario 2, where, compared to Scenario 1, all delivery due dates are shifted to the end of the week (cf. Table 2). This gives more leeway for job scheduling but makes it more important to balance energy costs against storage costs. It also serves as a plausibility check for the optimization algorithm. The cost function for Scenario 2 is shown in Figure 4 with the final oven and freezer allocations presented in Figure 6. For the oven allocation, it is noticeable that the jobs have been merged into two clusters and that, instead of pushing the two clusters together, a gap remains that is intended to take advantage of the reduced energy costs during the day (cf. energy price plot). Intuitively, without considering energy costs, the best solution would be to produce everything as late as possible in order to minimize storage costs. This is also the solution depicted in Figure 7, where, for validation, the same scenario has been tested with a modified cost function that excludes energy as a part goal. In particular, the coefficient ω_2 of f_{en} in Equation (1) was set to $\omega_2 = 0$.

Table 3 compares the results of Scenario 2 with and without energy, by taking the respective final solutions and evaluating them using the same cost function from Equation (1), including energy (i.e $\omega_2 = 2$). This makes sense because, either way, the energy costs still have to be paid in the end. This comparison shows that considering energy costs together with other production goals during optimization can result in an overall better solution. While the storage costs have slightly increased, this is outweighed by the gain in energy costs. Not considering energy costs and only optimizing storage costs would ignore significant potential for cost savings.

Table 3: Comparison between Scenario 2 with and without energy, showing final cost value using Equation (1) with $\omega_2 = 2$

	with Energy	**without Energy**	**Improvement**
Energy costs $\omega_2 \cdot f_{\text{en}}(x)$	173.53	188.92	8.15%
Storage costs $\omega_3 \cdot f_{\text{st}}(x)$	16.72	13.18	-26.86%
Total costs $f(x)$	**190.25**	**202.1**	**5.9%**

While the runtime performance of the simulation-based optimization has not yet been studied in detail, the case study gives a first indication: The scenarios with $n_{\text{iter}} = 100$ iterations needed about 2600-3600 evaluations (i.e. simulation runs), with one simulation run taking about 800 ms (1 week simulation time with 15 jobs). Overall, there is still room for decreasing the runtime by improving the implementation efficiency. However, these first results indicate feasible runtime also for larger scenarios.

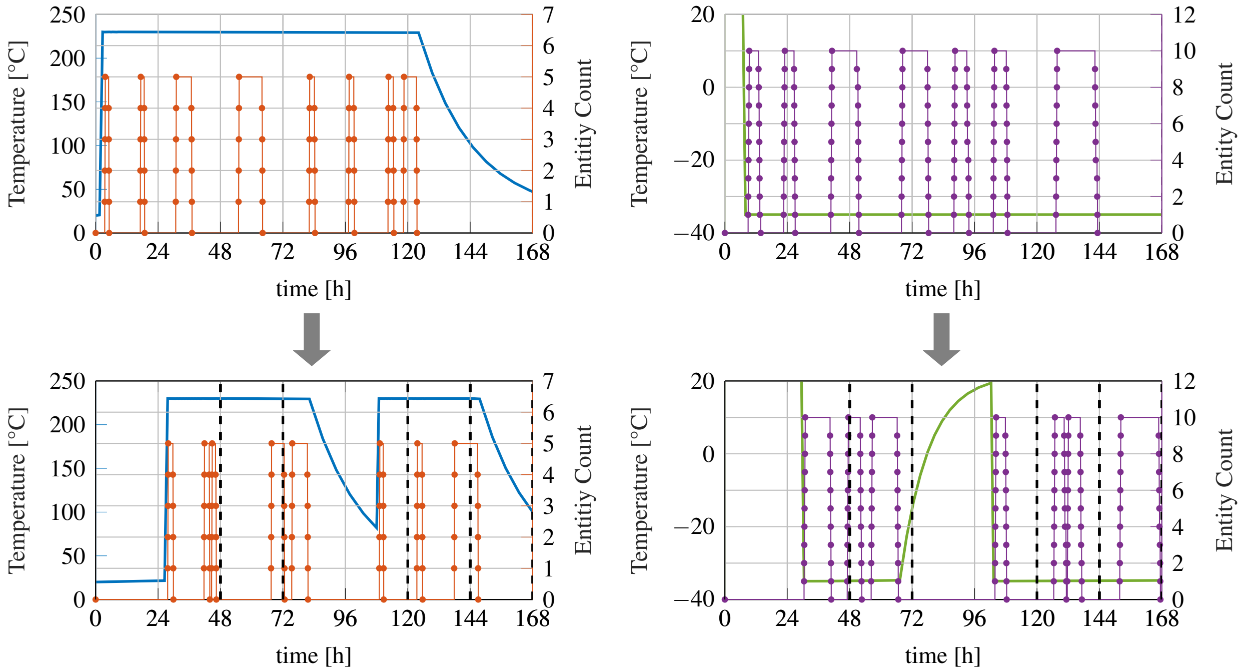

Figure 5: Optimization results for Scenario 1, initial solutions (top) and final results (bottom), both for oven (left) and freezer (right) allocation. Dashed vertical lines indicate job due times.

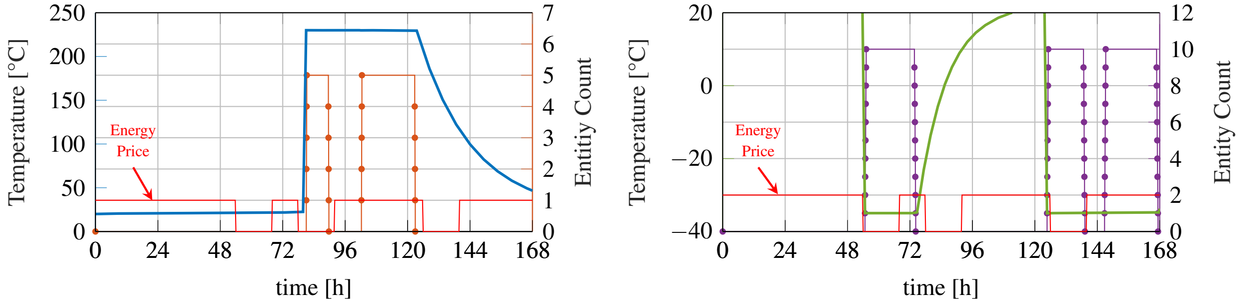

Figure 6: Optimization results for Scenario 2, showing allocation of oven (left) and freezer (right).

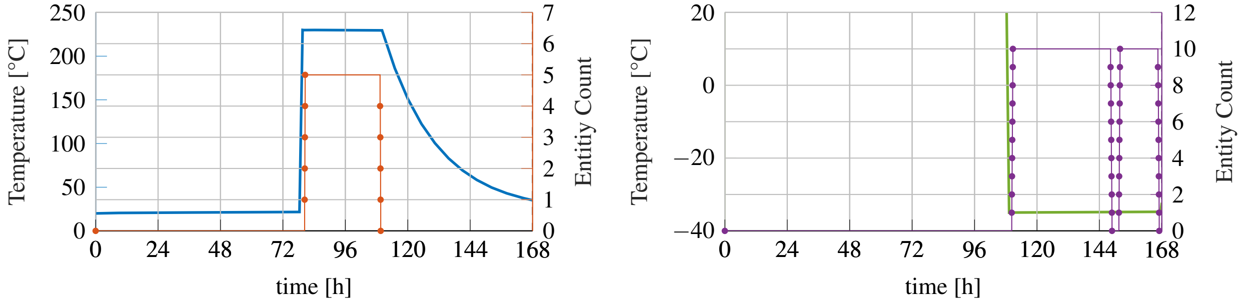

Figure 7: Optimization results for Scenario 2 without energy, showing allocation of oven (left) and freezer (right).

7 CONCLUSION

The case study serves as a demonstration that including energy considerations into the planning optimization can potentially reduce overall costs and therefore provide better planning results. Although the case study has been simplified to highlight the essential characteristics, it is easy to imagine applying the described method to more complex production systems with more diverse material flow and product types, as part of a practical PPC tool. This makes the optimization problem even more interesting because of the job completion times being more difficult to predict due to potential bottlenecks etc. From the current point of view, the presented simulation-based approach should also be applicable for these cases.

One advantage of the VNS metaheuristic is that it allows to tailor the operators and neighborhood structures to the individual problem instance, thereby allowing more efficient search and decreasing computation time. Neighborhoods (i.e. switching, shifting, etc.) are explored repeatedly and iteratively instead of sequentially, and the trade-off between exploration and exploitation can be controlled by the user. In contrast to population-based approaches, like Genetic Algorithms, VNS as a single-solution-based methods needs fewer function evaluations (i.e. simulation runs), which is a topic of importance for simulation-based methods. In the future, more extensive comparisons to population-based methods are planned. Experiments on the case study have also shown that the Simulated Annealing (SA) acceptance criterion is important especially for the merging operator, since, although merging two jobs might be beneficial in the end, there might sill be a short-term increase in the cost function (e.g. if the products from the second job are being produced sooner and therefore increase storage costs). In the future, VNS parameter and neighborhood calibration might still be improved and even be automated to directly adjust to the problem instance without the need for user intervention.

The hyPDEVS-based simulation itself, despite its hybrid nature, delivers sufficient performance to be feasible for simulation-based optimization tasks with with a large number of iterations. This has also been tested with larger real-world case studies, see e.g. (Sobottka, Kamhuber, Faezirad, and Sihn 2019).

ACKNOWLEDGMENTS

This work is funded by the Austrian Research Promotion Agency (FFG) as part of the project ASPeCT (Adaptive Smoothed Production, project number: 858655) within the research program "Production of the Future". The authors would like to thank all project partners for their contributions.

REFERENCES

Brailsford, S. C., T. Eldabi, M. Kunc, N. Mustafee, and A. F. Osorio. 2019, November. “Hybrid Simulation Modelling in Operational Research: A State-of-the-Art Review”. *European Journal of Operational Research* vol. 278 (3), pp. 721–737.

Chan, Y., R. Kantamaneni, and M. Allington. 2015, December. “Study on Energy Efficiency and Energy Saving Potential in Industry and on Possible Policy Mechanisms”. Technical report, ICF Cons. Ltd.

Deatcu, C., and T. Pawletta. 2012. “A Qualitative Comparison of Two Hybrid DEVS Approaches”. *SNE - Simulation Notes Europe* vol. 22 (1), pp. 15–24.

Freitas, A. A. 2004, December. “A Critical Review of Multi-Objective Optimization in Data Mining: A Position Paper”. *ACM SIGKDD Explorations Newsletter* vol. 6 (2), pp. 77–86.

Gansterer, M., C. Almeder, and R. F. Hartl. 2014, May. “Simulation-Based Optimization Methods for Setting Production Planning Parameters”. *International Journal of Production Economics* vol. 151.

Güller, M., Y. Uygun, and B. Noche. 2015, November. “Simulation-Based Optimization for a Capacitated Multi-Echelon Production-Inventory System”. *Journal of Simulation* vol. 9 (4), pp. 325–336.

Hansen, P., and N. Mladenović. 2009. “Variable Neighborhood Search Methods”. In *Encyclopedia of Optimization*, edited by C. A. Floudas and P. M. Pardalos, pp. 3975–3989. Boston, MA, Springer US.

Heinzl, B. 2016, November. *Hybrid Modeling of Production Systems: Co-Simulation and DEVS-Based Approach*. Diploma Thesis, TU Wien, Vienna, Austria.

Heinzl, B. 2020. *Methods for Hybrid Modeling and Simulation-Based Optimization in Energy-Aware Production Planning*. PhD Thesis, TU Wien, Wien.

Heinzl, B., and W. Kastner. 2019, December. “Platform-Independent Modeling for Simulation-Based Energy Optimization in Industrial Production”. *International Journal of Simulation: Systems, Science and Technology* vol. 20 (6), pp. 10.1–10.10.

Kamhuber, F., T. Sobottka, B. Heinzl, and W. Sihn. 2019, December. “An Efficient Multi-Objective Hybrid Simheuristic Approach for Advanced Rolling Horizon Production Planning”. In *Proceedings of the 2019 Winter Simulation Conference*, Volume 1, pp. 1–11. Maryland, USA, IEEE.

Kilger, C., H. Meyr, and H. Stadtler. 2015. *Supply Chain Management and Advanced Planning: Concepts, Models, Software, and Case Studies*. Springer.

Roshanaei, V., B. Naderi, F. Jolai, and M. Khalili. 2009, June. “A Variable Neighborhood Search for Job Shop Scheduling with Set-up Times to Minimize Makespan”. *Future Generation Computer Systems* vol. 25 (6), pp. 654–661.

Siarry, P. 2016, December. *Metaheuristics*. Springer.

Sobottka, T., F. Kamhuber, M. Faezirad, and W. Sihn. 2019. “Potential for Machine Learning in Optimized Production Planning with Hybrid Simulation”. *Procedia Manufacturing* vol. 39, pp. 1844–1853.

Swisher, J., P. Hyden, S. Jacobson, and L. Schruben. 2000, December. “A Survey of Simulation Optimization Techniques and Procedures”. In *Proceedings of the 2000 Winter Simulation Conference*, Volume 1, pp. 119–128. Orlando, FL, USA, IEEE.

Wari, E., and W. Zhu. 2016, September. “A Survey on Metaheuristics for Optimization in Food Manufacturing Industry”. *Applied Soft Computing* vol. 46, pp. 328–343.

Yazdani, M., M. Amiri, and M. Zandieh. 2010, January. “Flexible Job-Shop Scheduling with Parallel Variable Neighborhood Search Algorithm”. *Expert Systems with Applications* vol. 37 (1), pp. 678–687.

Zeigler, B. P., H. Prähofer, and T. G. Kim. 2000. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. USA, Academic Press.

AUTHOR BIOGRAPHIES

BERNHARD HEINZL received his Ph.D. from TU Wien and is currently a Research Assistant at the Institute of Computer Engineering at TU Wien. His research interests include hybrid modeling and simulation, model-driven engineering and metaheuristics in operations research. His email address is bernhard.heinzl@tuwien.ac.at.

WOLFGANG KASTNER is an Associate Professor at TU Wien and head of the Automation Systems Group at the Institute of Computer Engineering. His research focus lies in system integration in industrial as well as building automation. His email address is wolfgang.kastner@tuwien.ac.at.