Methods

Axel Schild*, Alexander Rose, Martin Grotjahn and Bennet Luck

A modular framework for optimal scheduling of industrial processes

Ein modularer Ansatz für die optimale Ausführungsplanung in industriellen Produktionsprozessen

https://doi.org/10.1515/auto-2021-0131
Received September 16, 2021; accepted November 19, 2021

Abstract: This paper proposes an extended Petri net formalism as a suitable language for composing optimal scheduling problems of industrial production processes with real and binary decision variables. The proposed approach is modular and scalable, as the overall process dynamics and constraints can be collected by parsing of all atomic elements of the net graph. To conclude, we demonstrate the use of this framework for modeling the moulding sand preparation process of a real foundry plant.

Keywords: Petri nets, MILP, optimal scheduling, batch process

Zusammenfassung: Dieser Beitrag schlägt einen erweiterten Petrinetz-Formalismus vor, der für die Zusammenstellung optimaler Planungsprobleme von vielfältigen industriellen Produktionsprozessen mit realen und binären Entscheidungsvariablen geeignet ist. Der vorgeschlagene Ansatz ist modular und skalierbar, da die Gesamtprozessdynamik und -beschränkungen durch Analyse und Betrachtung aller atomaren Elemente des Netzgraphen aufgestellt werden können. Der Ansatz wird abschließend für die Komposition des Planungsproblems des Formsandaufbereitungsprozesses einer realen Gießereianlage eingesetzt.

Schlagwörter: Petri-Netze, MILP, optimale Prozessplanung, Batch-Prozesse

***Corresponding author: Axel Schild,** IAV GmbH, Gifhorn, Germany, e-mail: axel.schild@iav.de
Alexander Rose, Martin Grotjahn, IKME – Advanced Control, Hannover University of Applied Sciences and Arts, Hannover, Germany, e-mails: alexander.rose@hs-hannover.de, martin.grotjahn@hs-hannover.de
Bennet Luck, IAV GmbH, Gifhorn, Germany, e-mail: bennet.Luck@iav.de

1 Introduction

Optimal scheduling of industrial production processes is a well and long investigated subject of the academic community as well as the by industrial practitioners [1, 3, 5, 6]. The task of optimal process/machine scheduling is often tackled sequentially by optimization or operations research experts, which first derive production unit models and afterwards synthesize the corresponding mixed-integer optimization problem by statement of a minimal performance metric as well as the minimal set of constraints that take the interdependence of production steps and assets into account. As this monolithic synthesis procedure itself requires deep process insight and optimization theory knowledge, it is not generic and, therefore, does not scale well when applied to all kinds of application domains as well as complex processes with many concurrency and synchronization elements or asynchronism. From an industrial perspective, it is a major obstacle that the resulting optimization problems are often very difficult to grasp and understand by anyone not involved in their synthesis, especially because such processes are typically subject to continuous adaptation and the availability of experts is typically low. Consequently, the majority of production processes are still scheduled by heuristic strategies, which are suboptimal in terms of both throughput and energy consumption. In order to accelerate the industrial adoption of optimal process scheduling, a generic, compositional framework is needed for both the modeling and for setting up the optimal scheduling problem. Such a framework must in particular ensure, that any adjustment of the process itself only requires a little effort for updating the optimization problem.

Modeling, simulation, and analysis of production processes have also been the subject of research for a long time. In the respective literature, process modeling is often performed using the powerful Petri net formalism [4, 7, 11], which provides compact, modular, and easy-to-understand models. In particular, process features like parallelism, concurrency and dependencies between subprocesses can be well captured and handled. Additionally, there exists a rich and established analysis theory for investigating the reachability set, liveliness and other important characteristics of processes. Petri nets are also used to devise all kinds of supervisory controllers [2, 10, 11], but they are seldom used in the context of optimal production process scheduling.

This paper contributes by combining the main concepts from the process analysis and the optimal process scheduling literature. It proposes a modular framework, which allows industrial practitioners to compose an optimal scheduling problem for a wide range of processes, without deep insight into optimization theory. Section 2 presents a small extension of the classical Petri net theory, which we believe is essential for capturing key properties, such as dwell time behavior found in a wide range of industrial production processes. For instance, it is common that subprocesses or process steps take different durations to finish and that actions like starting an electric drive must not be reverted before a minimum duration has passed. Also, subprocesses are often required to finish within a maximum duration after their start to meet quality requirements. Therefore, after a brief summary of Petri nets, two novel atomic elements, the *spontaneously forced transition* and the *spontaneously emptying transition*, are introduced. These new elements are in particular required to form the two novel net substructures we are looking for, namely the *minimum dwell-time place* and the *maximum dwell-time place*. Also, they are needed for capturing further process characteristics not possible with the standard elements. Section 3 continues with the explanation of how to construct the optimal scheduling problem in a bottom-up fashion by strictly considering local information of atomic elements. In section 4, we demonstrate the proposed approach at a real production process of a foundry plant. Section 5 concludes the paper with a summary and a short outlook on open issues.

2 An extended Petri net formalism for modeling industrial production processes

2.1 Definition of the extended Petri net formalism

In this publication, we adopt the notational conventions and definitions for *interpreted Petri nets* from [4] and extend these according to our needs for optimal

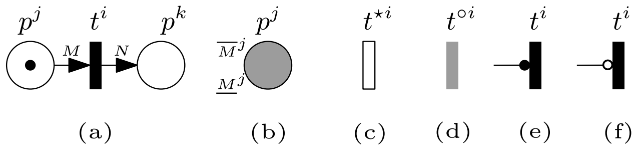

Figure 1: Atomic elements of a Petri net: (a) standard place-transition sequence with arc weighing, (b) buffer place with upper and lower capacities, (c) spontaneously forced transition, (d) spontaneous emptying transition, (e) test arc, (f) inhibitor arc.

process scheduling. A Petri net is bi-partite graph (ref. Fig. 1 (a)), which is formally defined by the tuple $\mathcal{PN} = \{\mathcal{P}, \mathcal{T}, \mathcal{P}pre, \mathcal{P}ost, \mathcal{M}_0, \mathrm{arc}_{\mathrm{pre}}, \mathrm{arc}_{\mathrm{post}}\}$ with sets of places $\mathcal{P} = \{p^0, \ldots, p^q\}$ and transitions $\mathcal{T} = \{t^0, \ldots, t^r\}$, $\mathcal{P}pre \subseteq \mathcal{P} \times \mathcal{T}$ denoting the set of incoming arcs, $\mathcal{P}ost \subseteq \mathcal{T} \times \mathcal{P}$ the set of outgoing arcs and $\mathcal{M}_0$ denoting the initial marking. The function $\mathrm{arc}_{\mathrm{pre}} : \mathcal{P} \times \mathcal{T} \to \mathbb{R}^+$ returns the arc weighting for incoming arcs $(p^j, t^i) \in \mathcal{P}pre$ and zero for all pairs $(p^j, t^i) \notin \mathcal{P}pre$. Likewise, the function $\mathrm{arc}_{\mathrm{post}} : \mathcal{T} \times \mathcal{P} \to \mathbb{R}^+$ returns the arc weighting for outgoing arcs $(t^i, p^j) \in \mathcal{P}post$ and zero for all pairs $(t^i, p^j) \notin \mathcal{P}post$. As an extension to classical theory, we allow for real and not only integer weightings, which ensure an efficient modeling of production process involving continuous educt quantities as common in the process industry.

We use $\bullet t^i = \{p^j, \ldots, p^k\} \subset \mathcal{P}$ to refer to pre-places, i. e., all places $p^j \mid \mathrm{arc}_{\mathrm{pre}}(p^j, t^i) > 0$ and $t^i \bullet = \{p^j, \ldots, p^k\} \subset \mathcal{P}$ to refer to post-places, for which $p^j \mid \mathrm{arc}_{\mathrm{post}}(t^i, p^j) > 0$. Similarly, we define $\bullet p^j$ and $p^j \bullet$ to refer to the pre- and post-transitions of a place p^j.

As a further extension to classical Petri net theory, the marking map $M : \mathcal{P} \to \mathbb{R}^+$ assigns a positive real value to each place p^j, which relates to the token loading. Moreover, each place may have a finite capacity, i. e., $0 \le \underline{M}_j \le M(p^j) \le \overline{M}_j \le \infty$ (see Fig. 1 (b)). Both extensions are required to efficiently model buffers, which are standard elements of any industrial production process. To improve the readability of net graphs, we use grey background color to indicate real-valued buffer places. Moreover, we indicate finite capacities by super- and subscripts on the left side of a place, but omit these in case of default lower and upper capacities 0 and 1. For optimal process scheduling, all Petri nets must evolve on a uniform time grid $T = (t_0, t_1, \ldots, t_N)$. We use $\mathcal{M}_k$ to refer to the marking of a Petri net at time k, which constitutes the state of a Petri net [4] and requires $p^j \in \mathcal{M}$, if and only if $M(p^j) > 0$. A *standard controlled transition* $t^i \in \{0, 1\}$ is called enabled, if and only if

$$\forall p^j \in \bullet t^i \mid M(p^j) - \mathrm{arc}_{\mathrm{pre}}(p^j, t^i) \cdot t^i \ge \underline{M}_j\,, \qquad (1)$$

$$\forall p^k \in t^i \bullet \mid M(p^k) + \mathrm{arc}_{\mathrm{post}}(t^i, p^k) \cdot t^i \le \overline{M}_k \qquad (2)$$

hold for $t^i = 1$. Note that (1), (2) only involve local information $\bullet t^i$ and $t^i \bullet$ from the immediate environment of t^i. When an enabled transition t^i is fired at time k, it updates the current marking $\mathcal{M}_k$ into its successor marking $\mathcal{M}_{k+1}$. This happens by simultaneously removing $\text{arc}_{\text{pre}}(p^j, t^i)$ tokens from all $p^j \in \bullet t^i$ and by generating $\text{arc}_{\text{post}}(t^i, p^k)$ new tokens in all $p^k \in t^i \bullet$. As indicated by the attribute, an enabled standard controlled transition may be fired by the supervisory controller, but it does not necessarily need to fire when conditions (1), (2) are fulfilled. Due to capacity constraints $M(p^j) \leq \overline{M}_j$ in force, we can actually omit (2) as these must hold for all k.

As a new element, complementary to standard controlled transitions, we introduce a *spontaneously forced transition* type denoted as $t^{\star i} \in \{0, 1\}$ (see Fig. 1 (c)). Such forced transitions may only be used within a restricted context: each transition $t^{\star i}$ must be connected to exactly one pre- and one post-place p^j and p^k and the upper capacities must fulfill $\overline{M}_k \geq \overline{M}_j$. In this context, the enabling and firing conditions become

$$\forall p^j \in \bullet t^i \mid M(p^j) - \text{arc}_{\text{pre}}(p^j, t^{\star i}) \cdot t^{\star i} \geq \underline{M}_j \,, \tag{3}$$

$$\forall p^j \in \bullet t^i \mid M(p^j) - \overline{M}_j \cdot t^{\star i} < \text{arc}_{\text{pre}}(p^j, t^{\star i}) \,. \tag{4}$$

As before, (3) is the enabling condition taking the marking of the pre-place into account and (4) is the forcing condition, which enforces a firing of t^i for as many update cycles as the number of tokens in p^j exceeds the weighting $\text{arc}_{\text{pre}}(t^i, p^k)$. We require such elements for composing the desired dwell-time places as shown in the next subsection. It is possible, to extend the forcing condition (4) to a larger modeling context involving synchronization structures, so that t_i possesses several pre-places. The necessary restriction to be imposed, then, is $\text{arc}_{\text{pre}}(p^j, t^i) = \overline{M}_j$.

Moreover, we introduce yet another transition type, referred to as the *spontaneously emptying transition* denoted by $t^{\circ i} \in \mathbb{R}^+$ (see Fig. 1 (d)), which is a real instead of a binary variable. Like spontaneously forced transitions, these elements may only be used in a specific modeling context: each transition $t^{\circ i}$ must be connected to exactly one pre- and one post-place p^j and p^k and the upper capacity $\overline{M}_k = \infty$ is infinite. In this context, the enabling and firing conditions become

$$\forall p^j \in \bullet t^i \mid M(p^j) - t^{\circ i} \geq \underline{M}_j \,, \tag{5}$$

$$\forall p^j \in \bullet t^i \mid M(p^j) - t^{\circ i} \leq \underline{M}_j \,. \tag{6}$$

As inferred from (5) and (6), a forced emptying transition will transfer all excessive tokens from the pre-place to the post-place at each update cycle of the net. Such transitions are required to capture the characteristics of transportation belts, which are also standard elements of industrial production processes.

In a composed Petri net, each previously introduced transition type may appear as a sink ($t^i \bullet = \emptyset$) or a source ($\bullet t^i = \emptyset$). The appearance of spontaneously forced sinks or sources (transition type (c) and (d)) are, therefore, only meaningful in conjunction with test or inhibitor arcs (see Fig. 1 (e) and (f)). Such arcs do not influence the token loading of their emerging place, but they require extensions to the set of enabling conditions. For a controlled transition, conditions

$$M(p^j) - t^i \geq 0 \,, \tag{7}$$

$$-M(p^j) - t^i \geq -1 \tag{8}$$

must be added, where (7) applies to a test arc and (8) to an inhibitor arc. For forced or emptying transitions, substructure-specific modifications to the enabling and forcing conditions are required rather than pure extensions.

2.2 Elementary net structures of complex Petri nets

Building arbitrarily complex nets $\mathcal{PN}$ by application of simple composition rules is possible with the atomic elements from subsection 2.1. Common basic substructures found in Petri nets are sequences, forks, choices, merge and synchronization elements [4]. These allow for modeling nearly any kind of complex industrial production process containing asynchronism, concurrency, parallelism, shared resources and synchronization elements. By aggregating substructures into user-defined places, complexity encapsulation and readability of net graphs can be further improved. In the following, we introduce two novel aggregated place types, the *minimum dwell-time place* and the *maximum dwell-time places* to capture the dwell-time behavior, which is a fundamental feature of almost any production process.

Proposition 1. *To enforce a minimum dwell-time behavior of N update cycles, a sequence of two places with finite capacity N, one place with capacity one and one forced transition is sufficient (see Fig. 2). The conditions associated with a minimum dwell-time place are*

$$-M(p^j) \geq -N \,, \tag{9}$$

$$-M(p^l) \geq -1 \,, \tag{10}$$

$$M(p^j) - t^m \geq 0 \,, \tag{11}$$

$$-M(p^j) + N \cdot t^m \geq 0 \,, \tag{12}$$

$$-M(p^n) \geq -N \,. \tag{13}$$

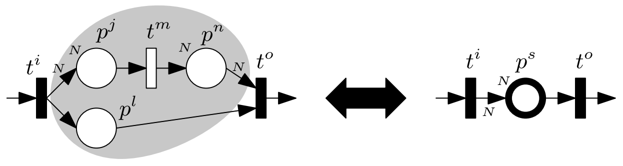

Figure 2: Minimum dwell-time structure to be aggregated into user-defined place.

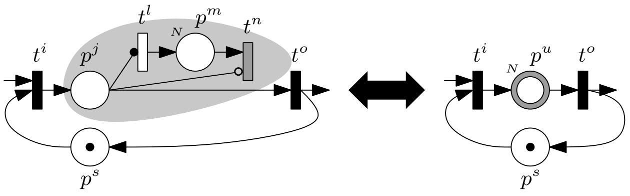

Figure 3: Maximum dwell-time structure to be aggregated into user-defined place.

Proof. Consider the situation, where firing of entry transition t^i at sample k adds N tokens to the place p^j. Condition (9) holds and prevents another firing of t^i until p^j and p^l are empty again. Conditions (10), (11) together require $t^m = 1$ for the sampling sequence $(k+1,\dots,k+N)$ and the number of tokens in p^j successively decrease. At sample time $(k+N+1)$, all tokens from p^j have been transfered to p^n. Condition (11) only holds for $t^m = 0$ and the exit transition t^o is finally enabled to fire $(N+1)$ time samples after the subprocess was started. As required for minimum dwell-time behavior, the process may reside more than N samples in this state. Due to (10), the predecessor transition t_i may also fire again earliest at the same time as t_o even though $p^j = 0$. Thus, the same subprocess may not be restarted before the aggregated place is emptied. □

Proposition 2. *To enforce a maximum dwell-time behavior of* $(N-1)$ *update cycles, a fork structure of two places, one with finite capacity N, one forced source transition, one emptying sink transition and a readiness-indicating place is sufficient (see Fig. 3). The conditions associated with the aggregated place are*

$$-M(p^j) + 1 \geq 0\,, \tag{14}$$
$$-M(p^m) + N \geq 0\,, \tag{15}$$
$$M(p^j) - t^l \geq 0\,, \tag{16}$$
$$-M(p^j) + t^l \geq 0\,, \tag{17}$$
$$-N \cdot p^j - t^n + N \geq 0\,, \tag{18}$$
$$M(p^m) - t^n \geq 0\,, \tag{19}$$
$$-M(p^m) + t^n + N \cdot M(p^j) \geq 0\,. \tag{20}$$

Proof. Consider the situation, where firing of the entry transition t^i at sample k adds one token to p^j and consumes the only token from p^s. Condition (16) holds and forces t^l to start generating tokens in p^m at the subsequent sample times until buffer p^m is finally full. Simultaneously, as long as $p^j = 1$, conditions (18) associated with the inhibitor arc prevents the emptying sink transition t^n to fire. Hence, t^o is forced to fire at sample $(k+N-1)$ at the latest, or otherwise (15) would become invalid at the next time instant $(k+N)$. This firing yields $p^j = 0$, so that conditions (18)–(20) force t^n to consume all tokens from p^m and reset the buffer. Also, $p^s = 1$, so that t^i may fire and restart the process again. If t^o is fired before $(k+N-1)$, then t^n empties p^m in the same manner, even though the maximum capacity is not reached. □

3 Composition of optimal scheduling problems

The equivalent state-space matrix notation of a net $\mathcal{PN}$ [4] allows us to easily set up an optimal scheduling problem

$$\min_{\boldsymbol{P},\boldsymbol{T}} J(\boldsymbol{p}_{k+1}, \boldsymbol{t}_k) = \sum_{k=0}^{H} \boldsymbol{w}^{\mathrm{T}}_{p(k+1)} \boldsymbol{p}_{k+1} + \boldsymbol{w}^{\mathrm{T}}_{tk} \boldsymbol{t}_k \text{ s.t.}\,, \tag{21}$$
$$\boldsymbol{p}_{k+1} = \boldsymbol{I}\boldsymbol{p}_k + \boldsymbol{N}\boldsymbol{t}_k,\ \forall k \in [0, H]\,, \tag{22}$$
$$0 \leq \boldsymbol{C}\boldsymbol{p}_k + \boldsymbol{D}\boldsymbol{t}_k + \boldsymbol{d},\ \forall k \in [0, H]\,, \tag{23}$$
$$0 \leq \boldsymbol{C}_{H+1}\boldsymbol{p}_{H+1}\,, \tag{24}$$

over the horizon length H, which constitutes a potentially large MILP that can be solved efficiently by many off-the-shelf solvers like CPLEX, GLKP, Gurobi and many others. In the above equations, $\boldsymbol{p}_k \in \mathbb{R}^{q,+}$ is a q-dimensional positive vector composed of elements $p_{j,k} = M(p^j)$ and $\boldsymbol{t}_k$ is a r-dimensional positive mixed-integer vector, where each element refers to the current value of the corresponding transition t^i. $\boldsymbol{I}$ denotes an $q \times q$ identity matrix and the $q \times r$ incidence matrix $\boldsymbol{N} = \boldsymbol{N}_{\text{post}} - \boldsymbol{N}_{\text{pre}}$ is the sum of $-\boldsymbol{N}_{\text{pre}}$ and $\boldsymbol{N}_{\text{post}}$ with matrix elements $n_{ji,\text{pre}} = \text{arc}_{\text{pre}}(p^j, t^i)$ and $n_{ki,\text{post}} = \text{arc}_{\text{post}}(t^i, p^k)$. The matrices $\boldsymbol{C} \in \mathbb{R}^{c\times q}$ and $\boldsymbol{D} \in \mathbb{R}^{c\times r}$ and the vector $\boldsymbol{d} \in \mathbb{R}^c$ aggregate all transition enabling and forcing conditions as well as place capacity limits. The collection of equations (22)–(24) can be conducted fully automatic by parsing each place and transition of a net, by picking the corresponding set from (1)–(8) and translating them into matrix notation. For each choice substructure contained in $\mathcal{PN}$, an additional mutual exclusion condition $\sum_{t^i \in p^j \bullet} t^i \leq 1$ must be added, to forbid the firing of multiple post-transitions of p^j.

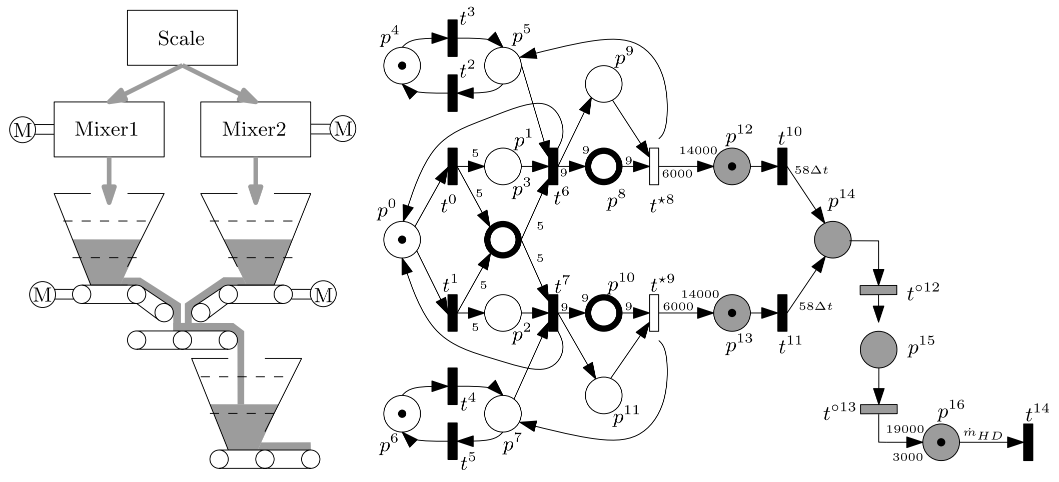
Scale
Mixer1
Mixer2
M
p^0 p^1 p^2 p^3 p^4 p^5 p^6 p^7 p^8 p^9 p^{10} p^{11} p^{12} p^{13} p^{14} p^{15} p^{16}
t^0 t^1 t^2 t^3 t^4 t^5 t^6 t^7 $t^{\star 8}$ $t^{\star 9}$ t^{10} t^{11} $t^{\circ 12}$ $t^{\circ 13}$ t^{14}
14000
6000
$58\Delta t$
19000
3000
$\dot{m}_{HD}$

Figure 4: Sand preparation of a foundry. left: schematic process; right: Petri net model.

After collecting the constraint set fully automatic, all that is left to define are the weighting vectors $\boldsymbol{w}_{pk} \in \mathbb{R}^q$ and $\boldsymbol{w}_{tk} \in \mathbb{R}^r$, which are typically time-invariant. The latter vectors $\boldsymbol{w}_{tk} \in \mathbb{R}^r$ are exclusively used to account for switching costs, while the former vectors can be used to penalize on-/off-times, process timing in general, or the inventory of buffers. Depending on the problem at hand, it may therefore be beneficial to introduce synthetic places into the process description, which are not necessarily needed for describing the process execution itself, but for the cost definition later on (counter places, timer places, etc.). By extending the decision variable space with slack variables, we can also impose costs for only specific variable intervals of p_k or soften capacity constraints.

4 An energy-optimal scheduling problem for a moulding sand foundry process

We have employed the previously introduced framework to set up the optimal scheduling problem for a moulding sand foundry process (see Fig. 4, left: schematic process). In this essential, fully automated production step of foundry plants, sand is prepared for the casting moulds. It starts by dosing a fixed amount $\Delta m_S = 6000$ kg of sand batch-wise into a scale and dumping this sand from the scale into one of two mixing units. In the mixing units, the sand is processed for a fixed duration to adjust its moulding properties. Afterwards, the processed sand is transported from the mixing unit hoppers to the machine hopper and buffered for consumption by the moulding process. The mixing units must be selected already at the beginning of the dosing. Both, the start of dosing as well as the dumping of the sand into the mixing units can be freely controlled. The dosing itself takes a finite duration of $\Delta t_S = 5\Delta t$, with Δt being the chosen cycle time of the Petri Net. Dumping of the sand into the mixing units happens instantaneously but is only permitted, if the mixing unit is running in idle. The sand processing takes a finite duration of $\Delta t_M = 9\Delta t$. The specification for the sand quality requires dumping the complete sand load Δm_S immediately at the end of processing into the corresponding hoppers. Both hoppers are equipped with a controllable extraction belt, which over each sampling interval can transfer each a finite mass $\Delta m_{MH} = 58\Delta t\, kg$ of sand onto a continuously running transport belt. The sand on the transport belt needs $\Delta t_T = 2\Delta t$ time units to arrive at the common machine hopper. On the downstream side, buffered sand is consumed from the machine hopper by the moulding machine with a fluctuating demand of Δm_{MD}. During production, the inventory of all three hoppers have to remain between lower and upper levels.

In contrast to our previous work [8, 9], where we followed the standard monolithic OCP construction approach, here we apply the previously introduced compositional Petri net formalism to set-up the process model and construct the optimal scheduling problem afterwards bottom-up. One advantage of this approach is, that we incorporate the process interdependencies directly into the model, which can be composed without any knowledge of mixed-integer linear programming (MILP), and do not have to worry about these any more, when setting up the constraint system. The resulting net graph is shown on the right side of Fig. 4. It consists of 16 places ($q = 16$), two of them *minimum dwell-time places* (see Fig. 2) and 5 buffer places, 14 transitions ($r = 14$), 10 standard controlled transitions, 2 spontaneously forced transitions (see Fig. 1, c) and 2 spontaneous emptying transitions (see Fig. 1, d), and 43 standard arcs. Thus without the introduced Petri net extensions, accurately capturing the essential properties of this still relatively simple process would have been impossible. All in all, we collect 33 constraints ($c = 33$), to describe the enabling and forcing conditions of the transitions. In addition to that, we have 32 box constraints resulting from the place capacity limits. Due to space constraints, we omit to state them here.

The scheduling objective of this process lies in minimizing the energy consumption, which is mainly related to the idle times of the mixing units ($w_{pk,5} = w_{pk,7} = 75\Delta t$) and costs for switching on the mixer motors ($w_{tk,3} = w_{tk,4} = 2025$). Moreover, temporary buffering of sand in the scale is penalized, which can be achieved by adding two additional constraints $p^3 - (1-p^1) - e^6 \leq 4$, $p^3 - (1-p^2) - e^7 \leq 4$ with continuous slack variables e^6 and e^7 as enabling indicators for the corresponding transitions and associate with these a cost of $w_{ek,i} = 10^6$. Moreover, we soften the hopper inventory constraints by introduction of four additional continuous slack variables s^{12u}, s^{13u}, s^{16l}, s^{16u}, and substitution of the corresponding hard inventory constraints.

In summary, we arrive at an optimal scheduling problem with 24 real and 12 binary decision variables per update cycle. Selecting a horizon length of $H = 240$ for a cycle time of $\Delta t = 10s$, this translates into a MILP with 5760 real and 2880 binary decision variables, 3840 equality and 15632 inequality constraints. We must emphasize, that the resulting mixed-integer optimization problem is not minimal in the number of decision variables and constraints, but the MILP is still tractable within seconds using CPLEX and state-of-the-art PC office hardware. This would be fast enough for deployment in the real plant.

5 Conclusions

This article proposes a new flavor of Petri nets, which are suited for modeling a large variety of production processes. The introduced substructures for imposing minimum and maximum dwell-time behavior are crucial elements for this task. It was explained, how the local information associated with each atomic element of a Petri net can be automatically collected and aggregated into a MILP optimization problem formulation. By solving such problems an optimal production schedule can be derived. The value of this concept was demonstrated by modeling a real-world moulding sand preparation process of a foundry plant, without the need to have knowledge about MILP.

Open issues to be further investigated are, how to improve numerical tractability, e. g., by condensing the usually non-minimal optimization problem resulting from the automatic synthesis process and the provision of an engineering toolbox, which automatically translates a process description into a supervisory control software for direct deployment on off-the-shelf embedded control hardware.

References

1. Doganis, P. and H. Sarimveis. 2007. Optimal scheduling in a yogurt production line based on mixed integer linear programming. *Journal of Food Engineering* 80(2): 445–453.
2. Karoui, O., Z. Li, N. Wu, M. Khalgui, E. A. Nasr and A. M. El-Tamimi. 2018. One-step control-ahead approach for the design of an optimal Petri-net based deadlock prevention policy. *IEEE Access* 6: 34307–34323.
3. Kondili, E., C. Pantelides and R. Sargent. 1993. A general algorithm for short-term scheduling of batch operations – i. milp formulation. *Computers & Chemical Engineering* 17(2): 211–227.
4. Lunze, J. 2017. *Ereignisdiskrete Systeme*. De Gruyter Oldenbourg.
5. Méndez, C., G. Henning and J. Cerdá. 2001. An milp continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. *Computers & Chemical Engineering* 25(4-6): 701–711.
6. Méndez, C. A., J. Cerdá, I. E. Grossmann, I. Harjunkoski and M. Fahl. 2006. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & chemical engineering* 30(6-7): 913–946.
7. Narahari, Y. and N. Viswanadham. 1985. A Petri net approach to the modelling and analysis of flexible manufacturing systems. *Annals of operations research* 3(8): 449–472.
8. Rose, A., M. Grotjahn, A. Schild and B. Luck. 2021. Application of feedback-corrected optimal scheduling for reducing the energy consumption of a mixing process in foundry. In: *Proceedings of IEEE Intern. Conf. on systems theory, control and computing*.
9. Rose, A., A. Schild, B. Luck and M. Grotjahn. 2021. Optimal control with linear integer programming for reducing the energy consumption of interdependent mixing machines in foundry. In: *2021 22nd IEEE International Conference on Industrial Technology (ICIT)*, vol. 1. IEEE, pp. 1138–1143.
10. Ye, J., Z. Li and A. Giua. 2014. Decentralized supervision of Petri nets with a coordinator. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45(6): 955–966.
11. Zhou, M. 1998. Modeling, analysis, simulation, scheduling, and control of semiconductor manufacturing systems: A Petri net approach. *IEEE Transactions on Semiconductor Manufacturing* 11(3): 333–357.

Bionotes

Axel Schild
IAV GmbH, Gifhorn, Germany
axel.schild@iav.de

Axel Schild received his Dr.-Ing. degree in Control Engineering in 2011 from Ruhr-Universität Bochum, Germany. Since then, he has held several industrial Senior Expert positions for advanced and predictive control across different industry sectors and applications. In this role, he initiated, coordinated and contributed to various publically-funded industrial-academic research consortia. His research activities focus on transfering latest-edge research results in the area of optimization and data-science into industrially-viable control solutions for energy and the automotive applications.

Alexander Rose
IKME – Advanced Control, Hannover University of Applied Sciences and Arts, Hannover, Germany
alexander.rose@hs-hannover.de

Alexander Rose received his master's degree in Mechatronics from Leibniz Universität Hannover in 2018, Germany. He is a PhD student at the Faculty of Mechanical Engineering at the Leibniz University Hannover and a member of the Advanced Control working group at the Institute of Engineering Design, Mechatronics and Electromobility at the Hannover University of Applied Sciences and Arts, Germany. His current research interests include development and application of optimal scheduling for energy intensive batch processes and data-based prediction of process variables with Long Short-Term Memory networks.

Martin Grotjahn
IKME – Advanced Control, Hannover University of Applied Sciences and Arts, Hannover, Germany
martin.grotjahn@hs-hannover.de

Martin Grotjahn received his Dr.-Ing. degree in the field of robotic control in 2003 from Leibniz University Hannover, Germany. Afterwards he led a team in the automotive industry that was responsible for the development of longitudinal dynamic control systems. In 2009, he became professor for Mechatronics at Hannover University of Applied Sciences and Arts, where he has held the position of Vice President for Research since 2020, following various leading positions in the Faculty of Mechanical Engineering and Bioprocess Engineering. He also heads the Advanced Control working group at the Institute of Engineering Design, Mechatronics and Electromobility. His main research interest is the transfer of control theory into real-world applications, e. g., in the fields of drive and automation technology, production technology and energy management of buildings.

Bennet Luck
IAV GmbH, Gifhorn, Germany
bennet.Luck@iav.de

Bennet Luck received his Dr.-Ing. degree in Control Engineering in 2016 from Gottfried Wilhelm Leibniz Universität Hannover, Germany. He currently heads the Control and Virtual Design Cluster at IAV. In this role, he initiated, coordinated and contributed to several publically-funded projects covering applied data science and advanced control solutions for industry processes and the applications in the energy sector.