Computers and Chemical Engineering 135 (2020) 106734

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Optimal production and maintenance scheduling for a multiproduct batch plant considering degradation

Ouyang Wu[a,b], Giancarlo Dalle Ave[c,d], Iiro Harjunkoski[c], Ala Bouaswaig[a], Stefan Marco Schneider[a], Matthias Roth[a], Lars Imsland[b,*]

[a] Automation Technology, BASF SE, Ludwigshafen 67056, Germany
[b] Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim 7491, Norway
[c] ABB Corporate Research Germany, Ladenburg 68526, Germany
[d] Process Dynamics and Operations Group, Department of Biochemical & Chemical Engineering, TU Dortmund, Dortmund 44221, Germany

ARTICLE INFO

Article history:
Received 18 September 2019
Revised 1 January 2020
Accepted 12 January 2020
Available online 13 January 2020

Keywords:
Maintenance scheduling
Sequence-dependent degradation
Performance decay
Symmetry breaking
Precedence models

ABSTRACT

Performance decay due to asset degradation is an important constraint in industrial production and therefore needs to be actively considered. This paper focuses on short-term scheduling for multiproduct batch processes with sequence-dependent degradation and is motivated by a case study in which the sequence of multiple-grade batch runs impacts evolution of fouling. A continuous-time scheduling formulation is proposed to incorporate realistic features of the case study processes. The precedence scheduling concept for the sequential process is employed to model sequences of multiproduct orders and maintenance and is implemented using the general disjunctive programming method. The scheduling formulation is applied to the case study and further analyzed through comprehensive computational tests, which illustrates the efficacy of the proposed formulation.

1. Introduction

The topic of joint production and maintenance scheduling has been popular in literature for decades. Industrial process units usually exhibit various types of degradation such as aging, fouling, wear, deformation, etc. (Martin et al., 1983; Bott, 1995; Zmitrowicz, 2006), which leads to performance decay during production and eventually becomes the termination criteria of production campaigns. In such a case, maintenance is required to reverse the degradation effects and needs to be considered while scheduling the production. Integration of production and maintenance in scheduling improves production performance when compared to the case where two problems are solved in a separate manner, in which preventive maintenance is performed to have a minimal effect on production (Dedopoulos and Shah, 1995). Therefore, degradation, being one of the main factors that affects production and maintenance, should be actively considered in scheduling. As a result, degradation models are built and employed to simulate the evolution of degradation, which are further integrated into optimization frameworks for scheduling. Various types of degradation have already been modeled in literature (Zhang et al., 1999; Teruel et al., 2005; Yeap et al., 2004; Wu et al., 2019a), and a review for degradation modeling approaches can be found in Gorjian et al. (2010). Accordingly, scheduling of production and maintenance are formulated using various types of mathematical programming formulations. Therefore, one challenge still remaining is to find appropriate scheduling formulations and degradation model structures that can be effectively combined to incorporate realistic processes features.

Recent examples from literature of production and maintenance scheduling are summarized as follows: Castro et al. (2014) presented a continuous-time model for maintenance scheduling in a gas engine power plant, which used multiple time grids and precedence-based sequencing variables for maintenance team and gas engine scheduling; the model was derived using generalized disjunctive programming (GDP) and compared both big-M and convex hull reformulations to find an appropriate set of constraints. Xenos et al. (2016) focused on optimal operation and maintenance scheduling in networks of compressors for chemical plants, where two types of washing procedures are considered with the aim of reducing extra power consumption due to fouling in the compressors; they proposed a discrete-time mathemat-

* Corresponding author
E-mail address: lars.imsland@ntnu.no (L. Imsland).

https://doi.org/10.1016/j.compchemeng.2020.106734

Nomenclature

Indices

i, i', i''	Production order
j	Unit
l	Stage
r	Batch recipe
v	Maintenance order
q	Break period for maintenance

Sets

I	Production orders
I_r	Production orders using recipe r
I_j^{aux}	Auxiliary order denoting the last batch run at unit j
J	Units
L	Stages
R	Batch recipes
V	Maintenance orders
Q	Break periods for maintenance
J_l	Subset of units belonging to stage l
J_d	Subset of units affected by degradation
L_d	Subset of stages containing units in J_d
L_{im}	Subset of stages that use an immediate precedence representation
L_g	Subset of stages that use a general precedence representation

Parameters

tp_{ij}	Fixed processing time of order i at unit j
ts_j	Start time when unit j becomes available
ts_{ql}	Start time of break period q in stage l
te_{ql}	End time of break period q in stage l
tm	Fixed time cost for maintenance at unit $j \in J_d$
tr_{il}	Time for material transfer of order i in stage l to the next stage
FI_j	Initial KPI value of fouling at unit j
Fc	KPI value after cleaning
R_{ri}	Recipe binary indicator for order i using recipe r
$A_{rj}, B_{rj}, A_{rj}^d, B_{rj}^d$	Fouling model parameters for recipe r at unit j (see Eqs. (1) and (2)
M, M_{**}^{**}	Big-M parameters
λ	Weight parameter
P_1, P_2	Breakpoints of piecewise function $G_j(\cdot)$
C_n^p	Slope parameters of segment $n \in \{1, 2, 3\}$ in $G_j(\cdot)$

Continuous variables

Ts_{il}	Start time of order i in stage l
Te_{il}	End time of order i in stage l
Ts_{vl}	Start time of order v in stage l
Te_{vl}	End time of order v in stage l
Tp_{il}	Processing time of order i in stage l
Tw_{il}	Waiting (idle) time of order i in stage l
$Tp_{k,j}$	Processing time of the k batch run at unit j
$f_{k,j}$	Fouling KPI value at the k batch run at unit j
f_{ij}	Fouling KPI value for order i at unit j
MS	Makespan
$f_{fi,j}$	Fouling KPI value at the last batch run at unit j
F_{jn}	Segment weight $n = \{1, 2, 3\}$ in piecewise function $G_j(f_{fi,j})$
G_j	Value of piecewise function $G_j(f_{fi,j})$

Binary variables

$X_{ii'l}$	Sequencing decision for order i immediately preceding order i' at a certain unit of stage l
X_{ivl}, X_{vil}	Sequencing decision for order i immediately preceding maintenance v or the other way round at a certain unit of stage l
Y_{ij}, Y_{vj}	Assignment decision of order i or maintenance order v at unit j
XF_{ij}, XF_{vj}	Sequencing decision of order i or maintenance order v in the first place of unit j
XL_{ij}	Sequencing decision of order i in the last place of unit j
$X_{ii'l}^g$	Sequencing decision for order i preceding order i' in stage l
Z_{il}	Decision of maintenance immediate before order i in stage $l \in L_d$
X_{iql}^g, X_{qil}^g	Sequencing decision for order i preceding breaking period q or the other way round in stage l
X_{vql}^g	Sequencing decision for maintenance v preceding breaking period q in stage l
X_{j1}^{pw}, X_{j2}^{pw}	Indicator showing $f_{fi,j}$ smaller than the corresponding breakpoint

ical model considering a condition-based maintenance model that minimizes the total operational costs as well as the wear of the compressors. Vieira et al. (2017) considered a bio-pharmaceutical manufacturing processes under performance decay, which has a maximum number of batches allowed per unit before maintenance; a Resource Task Network (RTN) continuous single-time grid formulation and bi-objective analysis are employed for production scheduling and maintenance planning towards several objectives such as profit maximization, minimization of the number of maintenance operations, etc. Biondi et al. (2017) proposed a multi-time-scale maintenance and production scheduling formulation using a discrete-time State Task Network representation; it investigated the effects of asset wear on the operation modes and maintenance with an example from steel industries and employed a metric of remaining useful life to keep track of the assets' life cycle. Aguirre and Papageorgiou (2018) proposed a medium-term continuous-time formulation for integrated production planning, scheduling and maintenance by considering multiple time periods; precedence models are employed to deal with sequence-dependent unit performance decay and flexible maintenance operation. Dalle Ave et al. (2019) considered electrode degradation in the context of demand side management (DSM) of stainless steel production and proposed an RTN-based discrete-time scheduling formulation to minimize both DSM-related costs and electrode replacement costs.

The state-of-the-art for batch scheduling is reviewed in Méndez et al. (2006) and Harjunkoski et al. (2014). Among those models, the precedence-based models as continuous-time representations have significantly lower number of variables and apply to sequential processes. In literature, Cerdá et al. (1997) firstly proposed unit-specific immediate precedence models for short-term scheduling of single-stage multiproduct batch plants. Méndez et al. (2000) and Gupta and Karimi (2003) presented an alternative formulation using the immediate precedence concept that improved computation efficiency and incorporated features for handling sequence-dependent setup times. On the other hand, the general precedence models proposed by Méndez et al. (2001) and Méndez and Cerdá (2003) extended the immediate precedence concept to all batches that precede another in the same sequence, which simplifies the model and reduces the number of sequencing variables. Besides, Castro and Grossmann (2012) presented linear generalized disjunctive programming (GDP) models for three different concepts of continuous-time representation to solve the short-term scheduling problem of single stage batch plants with parallel units. Compared with time-grid-based models that employ time-slots, periods, points or events, batch precedence models handle sequence-dependent activities straightforwardly and are effective given partial or total pre-ordering for production information (Harjunkoski et al., 2014).

This paper considers multiproduct batch processes that have product-specific degradation and provides an example of such a problem from a real industrial case study. The case study consists of a polymerization batch process that produces multiple-grade products. One major cause of degradation in this example is fouling in the reactors. While the monomer emulsion is polymerizing, produced polymers are accumulated in the inner surfaces of the units, especially in the heat exchangers of the reactor recirculation system. These residues cause reduced heat transfer from the product to the coolant thereby decreasing the total cooling efficiency resulting in a prolonged batch duration. Furthermore, the fouling increases flow resistance, resulting in increased pressure drops over the heat exchangers. If the polymer fouling has built-up an unacceptable level, the reaction section needs to be cleaned, and the reactor is shut down, disassembled, cleaned and assembled again. The cleaning occurs rather frequently in practice, once every few weeks, and takes several days to complete. In such a case, room for improvement remains in the realm of integrated scheduling of production and maintenance.

In this paper, a new scheduling formulation is proposed to solve a class of short-term scheduling and maintenance problems for multiproduct batch processes. The batch processes considered in this paper have sequence-dependent degradation, namely, the evolution of degradation in a unit is affected by the sequence of products which are produced in the said unit. Given orders of products, the scheduling algorithm makes sequencing and assignment decisions in a way that optimizes certain criteria, such as the minimization of makespan. The evolution of degradation and the corresponding performance decay are taken into account to schedule maintenance and production sequences. The scheduling problem is motivated by the aforementioned case study, in which a polymerization batch plant produces multiple grades of products and has fouling in the reaction section, and will be used as the test case for the proposed scheduling formulation.

The proposed formulation uses a continuous-time representation based on Wu et al. (2019b), which uses precedence models for the scheduling of multiproduct multistage batch plants and integrates degradation models to account for interactions between fouling evolution and batch operations. This paper extends the work of Wu et al. (2019b) by introducing symmetry-breaking constraints. In problems that include symmetries, one can reduce the size of the search space by eliminating symmetries. This is generally done by adding so-called symmetry-breaking constraints. Other works have also looked into symmetry breaking both for scheduling problems as well as optimization in general. Margot (2010) discussed symmetric integer linear programming problems, in which variables are permuted without changing the structure of the problem, and techniques that accelerate the solution algorithm via explicit handling the symmetry are reviewed. Erdirik-Dogan and Grossmann (2008) considered symmetry in a slot-based scheduling formulations due to identical units and derived the symmetry-breaking constraints for the case of four identical units. Baumann and Trautmann (2014) pointed out symmetry in a short-term scheduling formulation for make-and-pack production processes due to identical batches; symmetries were removed by imposing an arbitrary sequence for each group of identical batches leading to improved efficiency on computation cost. In the considered example, many batch orders of the same recipe are identical and switching the sequence of these identical batches does not make any difference to the operational performance. Corresponding symmetry-breaking constraints are proposed and extend (Baumann and Trautmann, 2014) by providing a pre-ordering of partial sets on precedence sequencing variables.

Another extension is integrating maintenance tasks with sequence-dependent degradation in the short-term scheduling formulation. Unlike the remaining useful life and the processing-time performance in Biondi et al. (2017) and Aguirre and Papageorgiou (2018), the motivating example takes a batch-to-batch process measurement as an indication of the degradation. Each fixed-size batch run, as well as its corresponding recipe type, and the current degradation level contribute to the evolution rate of degradation between batches, which shows dynamic degradation behaviour. This feature is incorporated into the precedence-based formulation through GDP models. While the production sequence contributes to the evolution of degradation, maintenance is considered in the formulation in order to reduce the degradation presented in a unit. Two types of representations for the maintenance tasks are introduced and integrated to the scheduling formulation via new GDP models that dictate the complex interactions between degradation, production, and maintenance operations.

This paper begins with a generic problem statement for the scheduling problem and process features that are considered as Section 2 presents. Next, the precedence-based mixed-integer linear programming (MILP) formulations are developed accounting for different features in the scheduling problem and is presented in detail in Section 3. The scheduling case study is further discussed in Section 4, while different MILP formulations are compared with the computational results being presented.

2. Problem statement

This section provides a generic problem statement for production and maintenance scheduling in multiproduct batch processes considering degradation.

Multistage and parallel units Multistage batch processes are considered in this paper, in which each stage $l \in L$ has parallel units $j \in J_l$. At least one of the stages is assumed to be affected by degradation denoted as $l \in L_d$, namely, the units affected by degradation are then denoted as $j \in J_d$.

Orders, recipes & maintenance tasks In batch production, orders $i \in I$ refer to requested batch products from higher level supply chain functions and must be assigned to one of the units in each stage for production. Timing variables Ts_{il}, Te_{il} denote the start and end time for processing order i in one of the units in stage l, respectively. Tp_{il} is the processing time of order i in stage l and can vary between units or unit conditions. Many orders can be of the same recipe $r \in R$, which makes those orders identical from the operational point of view. A maintenance task aims to restore degradation and is only processed at unit $j \in J_d$ in stage L_d and takes time tm to finish. Meanwhile, maintenance tasks are considered to be unavailable during certain periods because of resource constraints.

Sequence-dependent degradation Sequence-dependent degradation in parallel units of the batch plant is considered, and sequence information is used as inputs to model the evolution of degradation with a specific model structure. One example is fouling in batch reactors and external heat exchangers. Sequence-dependent fouling of the units is considered because the polymer grades and the reaction profiles affect the growth of the fouling from batch to batch. Wu et al. (2019c) developed a pressure-based key performance indicator (KPI) for indication of batch-to-batch fouling, where interfering factors due to batch production are excluded from the KPI. In Wu et al. (2019b), sequence-effects on the propagation of the fouling KPI is described using a linear model as

Eq. (1) presents.

$$f_{k+1,j} \quad = A_{R_{k,j}\ j} \cdot f_{k,j} + B_{R_{k,j}\ j} \tag{1}$$

where, $R_{k,j} = r$ indicates that the kth batch run at unit j uses recipe r; $f_{k,j}$ refers to the fouling KPI of the kth batch at unit j, which describes the degree of fouling at the beginning of the batch run. The fouling indicator thus follows recipe- and unit-relevant linear dynamics $\{(A_{rj}, B_{rj}) \mid r = 1, 2, ..., |R|,\ \ j = 1, 2, ..., |J_d|\}$. Furthermore, a maintenance task (cleaning) can be performed to restore the fouling KPI to a value Fc, and each unit that belongs to stage L_d has an initial fouling KPI value FI_j.

Performance decay Degradation and its effects on production are considered in different ways. Assuming a degradation KPI is known, which indicates the degree of degradation in each batch run, thresholds are needed to prevent a very high or low KPI value that could lead to production failures. Degradation can also lead to decay for production performance. In the example, the fouling KPI has a maximum threshold Fth to prevent from blockage in the heat exchangers, and fouling contributes to an increased reaction time, which is approximated using a linear structure from Wu et al. (2019b) as Eq. (2) shows:

$$Tp_{k,j} \quad = A^d_{R_{k,j}\ j} \cdot f_{k,j} + B^d_{R_{k,j}\ j},\ \ \forall\ j \in J_d \tag{2}$$

where, $Tp_{k,j}$ denotes the processing time of the kth batch run at unit j which depends on the fouling KPI and the types of recipes; $\{A^d_{rj},\ B^d_{rj}\}$ are the model parameters for orders with recipe r at unit j.

Material transfer & storage Material transfer operations are common in batch processes. We consider material transfer between two units from neighboring stages. In this case, it costs time tr_{il} and occupies both two units. We assume there is no intermediate storage between stages and unlimited storage tanks for final products.

3. Scheduling formulation

This section presents the MILP formulations for production and maintenance scheduling of multiproduct, multistage batch processes considering sequence-dependent degradation. Firstly, two continuous-time representations based on precedence concepts are reviewed. Next, novel constraints are proposed to handle sequence-dependent degradation, symmetries, maintenance in the continuous-time representations.

3.1. Continuous-time representations using precedence concepts

Precedence-based models are reviewed in Méndez et al. (2006) and Harjunkoski et al. (2014), and include both immediate precedence models and general precedence models.

3.1.1. Immediate precedence model

Following the immediate precedence concept proposed by Gupta and Karimi (2003), an order must immediately precede another order or it is the last one in one of the units of each stage as Fig. 1(a) shows. Alternatively, an order immediately follows another order or it is the first one in one of the units of each stage as Fig. 1(b) shows. The former cases of both representations are equivalent and have similar disjunction constraints, while the latter cases of both present no specific constraints on the timing of the first or the last order in a unit for parallel-unit processes. The corresponding disjunction constraints based on GDP method are presented in Eqs. (3) and (4).

$$\bigvee_{i' \neq i \in I} \begin{bmatrix} X_{i'il} \\ Te_{i'l} + Tw_{i'l} + tr_{i'l} \leqslant Ts_{il} \end{bmatrix} \bigvee_{j \in J_l} XF_{ij},\ \ \forall\ i \in I,\ \ l \in L \tag{3}$$

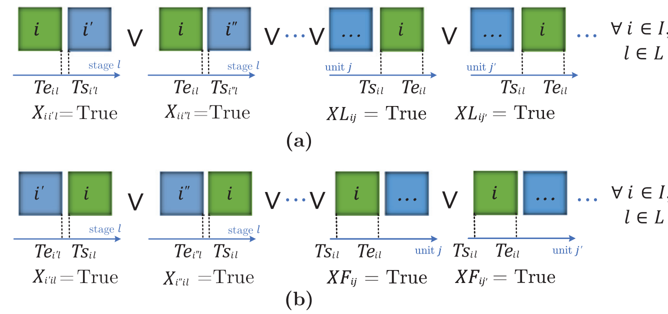

Fig. 1. Immediate precedence concept for scheduling (timing) in multistage parallel-unit batch plants with two representations in (a) and (b)

$$\bigvee_{i' \neq i \in I} \begin{bmatrix} X_{ii'l} \\ Te_{il} + Tw_{il} + tr_{il} \leqslant Ts_{i'l} \end{bmatrix} \bigvee_{j \in J_l} XL_{ij},\ \ \forall\ i \in I,\ \ l \in L \tag{4}$$

where, $X_{i'il} = 1$ indicates that order i is immediately processed after order i' at one unit of stage l; $XL_{ij} = 1$ indicates order i is the last one in the sequence of unit j, while $XF_{ij} = 1$ indicates the first order; disjunctions are the constraints connected by OR logical operator $\veebar$, where only a set of constraints has to be satisfied; Te_{il} is the end time of order i at one of the units in stage l and computed from the start time Ts_{il} as Eq. (5) shows; Tw_{il} denotes the waiting/idle time after processing order i in stage l allowing storage of order i in the current unit before transferring it to the next stage; tr_{il} denotes the time for material transfer of order i in stage l to the connection stage or storage tanks, which occupies both units or storage. The start time is always after the time when the corresponding unit becomes available as Eq. (6) shows.

$$Te_{il} = Ts_{il} + tr_{i(l-1)} + Tp_{il},\ \ \forall\ i \in I,\ \ l \in L \tag{5}$$

$$Ts_{il} \geqslant \sum_{j \in J_l} ts_j \cdot Y_{ij},\ \ \forall\ i \in I,\ \ l \in L \tag{6}$$

$$Ts_{i(l+1)} = Te_{il} + Tw_{il},\ \ \forall\ i \in I,\ \ l \in L : l < |L| \tag{7}$$

Tp_{il} in Eq. (5) is the total processing time of order i in stage l and depends on the sequencing and assignment binary variables such as $Tp_{il} = \sum_{j \in J_l} tp_{ij} \cdot Y_{ij}$, where tp_{ij} is the fixed processing time of order i at unit j; material transfer from upstream stage takes up the unit, and therefore $tr_{i(l-1)}$ is included to the processing of order i in stage l; tr_{i0} denotes the time for transferring raw materials to the first stage; Eq. (7) presents the timing of orders being processed in neighboring stages; since Eqs. (3) and (4) are equivalent, the logic constraints in Eq. (3) are taken and reformulated into MILP constraints using the big-M method as Eq. (8) shows:

$$Te_{i'l} + Tw_{i'l} + tr_{i'l} \leqslant Ts_{il} + M \cdot (1 - X_{i'il}),\ \ \forall\ i, i' \in I : i \neq i',\ \ l \in L \tag{8}$$

Further constraints are derived from GDP statements based on Fig. 1: the constraint in Eq. (9) puts orders either as the first one in the sequence or after another order of a unit in each stage; the constraint in Eq. (10) assigns only one order to the first place of the sequence at each unit; the constraint in Eq. (11) assigns orders either as the last one or immediately before another order in the sequence of any units; the constraint in Eq. (12) assigns only one order to the last place of the sequence at each unit. Other sequencing and assignments constraints for immediate precedence models are presented in Eqs. (13)–(15), which assigns orders to one of the units in each stage including the first and last place of units; the constraint in Eq. (16) assigns consecutive orders to one unit (Gupta and Karimi, 2003). Moreover, Cerdá et al. (1997) mentioned

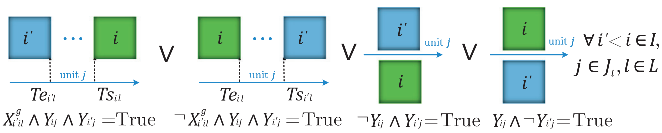

Fig. 2. General precedence concept for scheduling (timing) in multistage parallel-unit batch plants

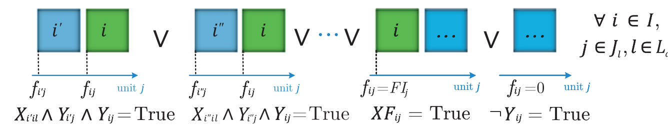

Fig. 3. Immediate precedence concept for sequence-dependent fouling evolution in multistage parallel-unit batch plants

computational improvement of the immediate precedence model by including redundant constraints, which avoids subtours on sequences as Eqs. (17) and (18) show.

$$\sum_{j\in J_l} XF_{ij} + \sum_{i'\in I: i'\neq i} X_{i'il} = 1, \forall\ i\in I,\ \ l\in L \tag{9}$$

$$\sum_{i\in I} XF_{ij} \leqslant 1, \forall\ j\in J_l,\ \ l\in L \tag{10}$$

$$\sum_{i'\in I: i'\neq i} X_{ii'l} + \sum_{j\in J_l} XL_{ij} = 1, \forall\ i\in I,\ \ l\in L \tag{11}$$

$$\sum_{i\in I} XL_{ij} \leqslant 1, \forall\ j\in J_l,\ \ l\in L \tag{12}$$

$$\sum_{j\in J_l} Y_{ij} = 1, \forall\ i\in I,\ \ l\in L \tag{13}$$

$$Y_{ij} \geqslant XF_{ij}, \forall\ i\in I,\ \ j\in J_l,\ \ l\in L \tag{14}$$

$$Y_{ij} \geqslant XL_{ij}, \forall\ i\in I,\ \ j\in J_l,\ \ l\in L \tag{15}$$

$$Y_{ij} \leqslant Y_{i'j} + 1 - X_{ii'l} - X_{i'il},\ \ \forall\ i, i'\in I : i\neq i',\ \ j\in J_l,\ \ l\in L \tag{16}$$

$$X_{i'il} + X_{ii'l} \leqslant 1, \forall\ i, i'\in I : i\neq i',\ \ l\in L \tag{17}$$

$$X_{i'il} + X_{ii''l} + X_{i''i'l} \leqslant 2, \forall\ i, i', i''\in I : i\neq i'\neq i'',\ \ l\in L \tag{18}$$

3.1.2. General precedence model

Comparing with immediate precedence models, general precedence models extend the local precedence of two orders to a global precedence, in which the sequencing representation of any two orders are presented as three cases: one order precedes another at a specific unit; the opposite holds true and the order follows the second; the third case is that two orders are assigned to different units as Fig. 2 illustrates. The corresponding GDP-based constraints for the timing of any two orders are presented in Eq. (19):

$$\begin{bmatrix} X^g_{i'il}\wedge Y_{ij}\wedge Y_{i'j} \\ Te_{i'l} + Tw_{i'l} + tr_{i'l} \leqslant Ts_{il}\end{bmatrix} \underline{\bigvee} \begin{bmatrix} \neg X^g_{i'il}\wedge Y_{ij}\wedge Y_{i'j} \\ Te_{il} + Tw_{il} + tr_{il} \leqslant Ts_{i'l}\end{bmatrix},$$
$$\forall\ i, i'\in I : i' < i,\ \ j\in J_l,\ \ l\in L \tag{19}$$

where, $X^g_{i'il}\wedge Y_{ij}\wedge Y_{i'j} = 1$ indicates order i precedes order i' at unit j of stage l, while $\neg X^g_{i'il}\wedge Y_{ij}\wedge Y_{i'j} = 1$ indicates order i' precedes order i at unit j of stage l; the two types of disjunctions correspond to the timing constraints for the first two cases, while the third case presents no constraints for the timing of the two order and therefore is omitted for brevity. The big-M reformulated constraints are presented in Eq. (20). Moreover, the general precedence models share some constraints with the immediate precedence models, which are Eqs. (5)–(7) and (13).

$$\begin{cases} Te_{i'l} + Tw_{i'l} + tr_{i'l} \leqslant Ts_{il} + M\cdot(3 - X^g_{i'il} - Y_{ij} - Y_{i'j}), \\ Te_{il} + Tw_{il} + tr_{il} \leqslant Ts_{i'l} + M\cdot(2 + X^g_{i'il} - Y_{ij} - Y_{i'j}),\end{cases}$$
$$\forall\ i, i'\in I : i' < i,\ \ j\in J_l,\ \ l\in L \tag{20}$$

3.2. Constraints on the scheduling problem based on precedence concepts

The standard precedence models were discussed in the previous subsection, which forms the basic continuous-time representation for the scheduling problem. In this subsection, new constraints are introduced to the formulation to consider more problem-specific aspects including sequence-dependent degradation and symmetry-issues related to production sequences. Constraints for sequence-dependent degradation are presented and obtained using the GDP method and the big-M reformulation, which rely on immediate precedence variables for the sequencing representation. Symmetry from identical order sequences is recognized as a computational problem and therefore also investigated. New constraints are proposed to break the symmetry so that the computational performance of the formulation is improved.

3.2.1. Sequence-dependent degradation

Sequence-dependent degradation is considered and relates to different aspects of the scheduling problem. In the case study, the production sequence affects the batch-to-batch fouling evolution. It is further described using a recipe-dependent linear first-order structure as Eq. (1) presents, that is, the fouling KPI in the kth batch propagates from the one in the immediate previous batch, and the corresponding batch recipe contributes to its propagation. To represent the fouling evolution model in the scheduling problem, Wu et al. (2019b) introduced a new notation f_{ij} as replacement of $f_{k,j}$ and integrated it into an immediate precedence model. The sequence-dependent feature of the fouling evolution is coupled to the immediate precedence of the orders, from which a set of disjunctions as Fig. 3 illustrates are derived. It shows three types of occasions for a particular order and a unit: order i is preceded by another batch order at unit j, order i is the first one in the sequence of unit j or it is assigned to a different unit from unit j. The logic constraints for these scenarios are presented in Eq. (21):

$$\underline{\bigvee}_{i'\in I: i'\neq i}\begin{bmatrix} X_{i'il}\wedge Y_{i'j}\wedge Y_{ij} \\ f_{ij} = AS_{i'j}\cdot f_{i'j} + BS_{i'j}\end{bmatrix} \underline{\bigvee} \begin{bmatrix} XF_{ij} \\ f_{ij} = FI_j\end{bmatrix} \underline{\bigvee} \begin{bmatrix} \neg Y_{ij} \\ f_{ij} = 0\end{bmatrix},$$
$$\forall\ i\in I,\ \ l\in L_d,\ \ j\in J_l \tag{21}$$

where, f_{ij} is the fouling KPI of order i at unit j and calculated from the unit-specific fouling evolution model in Eq. (1) if order i is processed at unit j and immediately preceded by another order ($X_{i'il}\wedge Y_{i'j}\wedge Y_{ij} = 1$); the recipe-dependent parameters of the model are defined as $AS_{ij} = \sum_{r\in R} R_{ri}A_{rj}$, $BS_{ij} = \sum_{r\in R} R_{ri}B_{rj}$, in which the binary variable R_{ri} indicates whether order i is using recipe r; if order i is the first one in the sequence of unit j ($XF_{ij} = 1$), f_{ij} equals to the initial fouling value FI_j of unit j; additionally f_{ij} is set to be zero if order i is not assigned to unit j ($\neg Y_{ij} = 1$), and this makes $f_{ij} = 0$ the same as $Y_{ij} = 0$ on indicating that order i is not assigned to unit j.

The processing time of an individual batch run in a degraded unit varies and is calculated based on the corresponding fouling KPI as Eq. (2) presents. To represent Eq. (2) using variables from the scheduling formulation, both the binary variable Y_{ij} and the continuous variable f_{ij} are employed as they indicate whether order i is assigned to unit j ($Y_{ij} = 1,\ \ f_{ij}\neq 0$) or not ($Y_{ij} = 0,\ \ f_{ij} = 0$).

As a result, Tp_{il} can be calculated through summation of the terms that multiply the processing time of order i in each unit according to Eq. (2) with the corresponding unit assignment variable over all units of stage l, which is presented in Eq. (22). The additional dimension of f_{ij} showing unit assignment leads to a linear constraint for the calculation of Tp_{il}.

$$Tp_{il} = \sum_{j \in J_l} \sum_{r \in R} (R_{ri} \cdot A^d_{rj} \cdot f_{ij} + R_{ri} \cdot B^d_{rj} \cdot Y_{ij}), \ \ \forall i \in I, l \in l_d \tag{22}$$

The logic expressions in Eq. (21) are reformulated to be part of MILP using the big-M method, and the big-M constraints are presented in Eqs. (23)–(25).

$$\begin{cases} f_{ij} \geqslant AS_{i'j} \cdot f_{i'j} + BS_{i'j} - M^{26}_{i'ij} \cdot (3 - X_{i'il} - Y_{i'j} - Y_{ij}), \\ f_{ij} \leqslant AS_{i'j} \cdot f_{i'j} + BS_{i'j} + M^{27}_{i'ij} \cdot (3 - X_{i'il} - Y_{i'j} - Y_{ij}), \end{cases}$$
$$\forall \ i, i' \in I : i' \neq i, \ \ l \in L, \ \ j \in J_l \tag{23}$$

$$\begin{cases} f_{ij} \geqslant FI_j - M^{28}_{ij} \cdot (1 - XF_{ij}), \\ f_{ij} \leqslant FI_j + M^{29}_{ij} \cdot (1 - XF_{ij}), \end{cases} \ \ \forall \ i \in I, \ \ l \in L, \ \ j \in J_l \tag{24}$$

$$f_{ij} \leqslant 0 + M^{30}_{ij} \cdot (Y_{ij}), \qquad \forall \ i \in I, \ \ l \in L, \ \ j \in J_l \tag{25}$$

where, the equality constraints in the logic expressions result in two types of inequality constraints by adding the big-M terms as Eqs. (23) and (24) show, while the counterpart of Eq. (25) is redundant and is removed from the MILP since f_{ij}'s lower bound is zero in the example. The values of big-M coefficients depend on the lower-bound and upper-bound of f_{ij}. As a result, these coefficients such as $M^{26}_{i'ij}$ are calculated to get the tightest values as Eqs. (26)–(30) illustrate.

$$M^{26}_{i'ij} = \max(AS_{i'j} \cdot f_{i'j} + BS_{i'j} - f_{ij}), \forall \ i \neq i' \in I, \ \ l \in L, \ \ j \in J_l \tag{26}$$

$$M^{27}_{i'ij} = \max(f_{ij} - AS_{i'j} \cdot f_{i'j} - BS_{i'j}), \forall \ i \neq i' \in I, \ \ l \in L, \ \ j \in J_l \tag{27}$$

$$M^{28}_{ij} = \max(FI_j - f_{ij}), \forall \ i \in I, \ \ l \in L, \ \ j \in J_l \tag{28}$$

$$M^{29}_{ij} = \max(f_{ij} - FI_j), \forall \ i \in I, \ \ l \in L, \ \ j \in J_l \tag{29}$$

$$M^{30}_{ij} = \max(f_{ij}), \forall \ i \in I, \ \ l \in L, \ \ j \in J_l \tag{30}$$

3.2.2. Symmetry-breaking constraints

In the considered case study, batch orders that use the same recipe are identical. As a result, the production sequences composed of recipe-nonspecific indexes of batch orders are equivalent on production performance when switching identical batches as solutions 1–3 show in Fig. 4, in which the color of the batch order denotes the recipe and the product grade. Symmetry in the scheduling problem formulation refers to those equivalent solutions. Ruling out those equivalent solutions will break the production sequence symmetries and improve the computational efficiency of the optimization process.

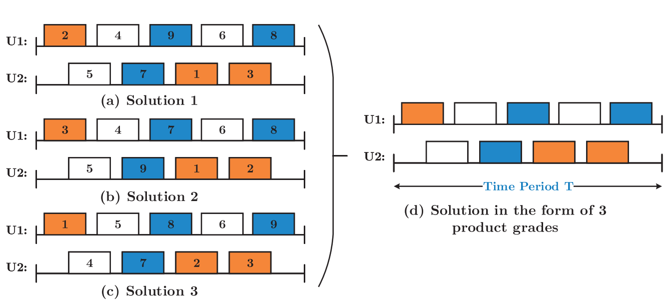

Fig. 4. Symmetry in the production sequence: nine batch orders with three product grades denoted by different colors are scheduled in two units

The strategy to break the symmetries that result from identical batch orders is to enforce a predefined sequence for each subgroup of batch orders that use the same recipe. This predefined subgroup sequence is based on index values, and identical batch orders are arranged in ascending order for each stage as solution 3 shows in Fig. 4. Each solution of the scheduling problem that follows the predefined subgroup sequences belongs to a symmetry group that has all the equivalent solutions, and those equivalent solutions are excluded from the feasible region of the scheduling problem without ruling out the possible optimal solution by forcing the subgroup sequences.

Constraints that enforce those predefined subgroup sequences on the scheduling problem are the so-called symmetry-breaking constraints. One of the constraints cuts the solution space of continuous variables such as the timing variables for each batch order. This constraint illustrated in Eq. (31) as proposed in Baumann and Trautmann (2014) enforces the start time of batches to follow the predefined subgroup sequence. The second type of symmetry-breaking constraints cuts off symmetry solutions by fixing certain specific precedence binary variables, and therefore the potential number of branch-and-bound tree nodes declines which in turn speeds up solving the MILP scheduling problem. Taking general precedence models as example, the precedence variables are fixed to one when order i that precedes another identical order i' has a smaller index value as Eq. (32) shows. In immediate precedence models, specific precedence binary variables are fixed to zeros to prevent a batch with larger index value from being processed right before another identical batch in the same unit as Eq. (33) illustrates. Besides, following the subgroup sequences only allows some of those identical orders that have smaller index value to be put at the first place of the production sequence in one of the units ($XF_{ij} = 1$). The number of the smaller-indexed batches is related to the number of units in a stage. For example, it is possible to have one of the two batches with smaller indexes in a recipe group to be first placed in the unit for the stage with two paralleled units. The generalized symmetry-breaking constraint on XF_{ij} is presented in Eq. (34).

$$Ts_{il} \leqslant Ts_{i'l}, \ \ \forall \ i, i' \in I_r : i < i', \ \ r \in R, \ \ l \in L \tag{31}$$

$$X^g_{ii'l} = 1, \ \ \forall \ i, i' \in I_r : i < i', \ \ r \in R, \ \ l \in L_g \tag{32}$$

$$X_{i'il} = 0, \ \ \forall \ i, i' \in I_r : i < i', \ \ r \in R, \ \ l \in L_{im} \tag{33}$$

$$\sum_{r=1}^{|R|} \sum_{i=min(I_r)}^{i+n-1} XF_{ij} \leqslant 1, \ \ \forall \ j \in J_l, \ \ l \in L_{im} \tag{34}$$

where I_r denotes the set of identical batches using recipe r; L_g denotes the set of the stages that use the general precedent representation, and L_{im} denotes the set of the stages that use the immediate precedence representation.

These definitions of predefined subgroup sequences can cause contradiction in certain situations between different stages. Eq. (31) describes the subgroup sequences according to start time, but it is trivial to determine the sequence of two batch orders in different units. If the processing time of those identical batches are unit-specific, the subgroup sequence defined by start time can be different from the one defined by end time as the scenario in Fig. 5 shows. In Fig. 5 (1), order 2 starts later but ends earlier than order 1 in stage L_1 and follows the subgroup sequence defined by start time, but order 2 has to wait until completion

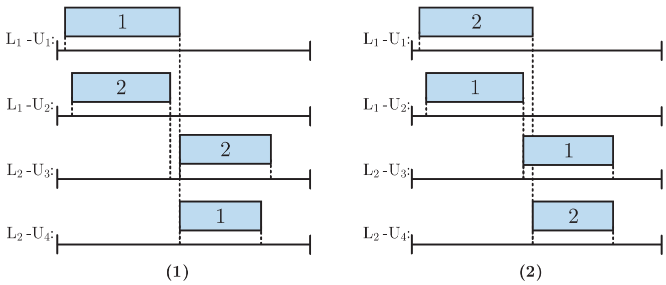

Fig. 5. Symmetry-breaking constraints for two stage two parallel units

of order 1 in stage L_2 to follow the corresponding subgroup sequence and therefore takes more time. In Fig. 5 (2), the subgroup sequence in stage L_1 is defined by end time of orders, while the one in stage L_2 is defined by the start time. This presents a better solution with less production time without breaking the predefined subgroup sequences in two stages. Therefore, the strategy to deal with symmetry in scheduling of multistage and parallel-unit processes is applying symmetry-breaking constraints only on one stage or two neighboring stages; for symmetry-breaking constraints in neighboring stages, it is recommended to use the end time of orders for indicating subgroup sequences in the upstream stage and the start time of orders in the downstream stage.

3.3. Scheduling of maintenance operations

Maintenance operations are required to reverse the degradation which occurs in batch production and therefore need to be explicitly considered in the scheduling formulation. In the considered example, maintenance refers to the cleaning that removes fouling in the reactors and heat exchangers. It shuts down the batch units for a period of time and restores the fouling KPI to a non-fouled value *Fc*. Two types of formulations are presented for the scheduling of maintenance operations in the units that have sequence-dependent fouling.

3.3.1. C1: maintenance operations as special orders

This formulation takes maintenance additionally as a special type of order denoted as $v \in V$, which applies to the units that have sequence-dependent fouling ($j \in J_d$) and belong to a specific stage ($l \in L_d$) in the case study example. The number of maintenance orders $|V|$ is usually far less than the number of production orders $|I|$ in the horizon of short-term scheduling, and it is not always necessary to perform maintenance when the value of the fouling KPI is still at an acceptable level.

The binary variables for maintenance orders in the immediate precedence models are introduced, and new constraints or modifications to the constraints are presented in order to integrate maintenance orders in the scheduling formulation. The sequencing variables $X_{ivl} = 1$ or $X_{vil} = 1$ indicate order i precedes order v immediately or the converse at unit assigned with order i in stage $l \in L_d$; similarly, $Y_{vj} = 1$ means order v is undertaken at unit j. Following the immediate precedence concept among maintenance and production orders in stage $l \in L_d$, the GDP-based constraints for the timing of two different types of orders are obtained and presented in Eq. (35). The corresponding big-M constraints are derived as Eq. (36) shows.

$$\begin{bmatrix} X_{ivl} \\ Te_{il} + Tw_{il} + tr_{il} \leqslant Ts_{vl} \end{bmatrix} \underline{\bigvee} \begin{bmatrix} X_{vil} \\ Te_{vl} + Tw_{vl} \leqslant Ts_{il} \end{bmatrix}, \quad \forall i \in I,\ v \in V, j \in J_l,\ l \in L_d \tag{35}$$

$$\begin{cases} Te_{il} + Tw_{il} + tr_{il} \leqslant Ts_{vl} + M \cdot (1 - X_{ivl}), \\ Te_{vl} + Tw_{vl} \leqslant Ts_{il} + M \cdot (1 - X_{vil}), \end{cases} \quad \forall i \in I,\ v \in V, j \in J_l,\ l \in L_d \tag{36}$$

where, Te_{vl} and Ts_{vl} denote the start and end time of maintenance order v that takes time tm and are linked via Eq. (37).

$$Te_{vl} = Ts_{vl} + tm,\ \forall\ v \in V, l \in L_d \tag{37}$$

The logic constraint in Eq. (38) accounts for the effect of maintenance tasks on the fouling KPI, which performs as supplement to Eq. (21).

$$\bigvee_{i \in I, v \in V, j \in J_l, l \in L_d} \begin{bmatrix} X_{vil} \wedge Y_{ij} \\ f_{ij} = F_c \end{bmatrix} \tag{38}$$

The big-M constraints are presented in Eq. (39), while the corresponding specific big-M coefficients are obtained in Eqs. (40) and (41).

$$\begin{cases} f_{ij} \geqslant F_c - M_{ij}^{40} \cdot (2 - X_{vil} - Y_{ij}), \\ f_{ij} \leqslant F_c + M_{ij}^{41} \cdot (2 - X_{vil} - Y_{ij}), \end{cases} \quad \forall\ i \in I,\ v \in V,\ j \in J_l,\ l \in L_d \tag{39}$$

$$M_{ij}^{40} = \max(F_c - f_{ij}),\ \forall\ i \in I,\ v \in V,\ j \in J_l,\ l \in L_d \tag{40}$$

$$M_{ij}^{41} = \max(f_{ij} - F_c),\ \forall\ i \in I,\ v \in V,\ j \in J_l,\ l \in L_d \tag{41}$$

The constraint in Eq. (42) allows each maintenance order to be undertaken in one of the units or not occur at all. The constraints in Eqs. (9) and (10) from the immediate precedence model are extended to sequence maintenance orders among production orders as Eqs. (43)–(45) show. Similarly the constraint in Eq. (11) is replaced with new ones in Eqs. (46) and (47). In addition, the constraints in Eqs. (48) and (49) are supplements to the constraint in Eq. (16) for assigning both production and maintenance orders consecutively to one unit.

$$\sum_{j \in J_l} Y_{vj} \leqslant 1,\ \forall\ v \in V,\ l \in L_d \tag{42}$$

$$\sum_{i \in I} XF_{ij} + \sum_{v \in V} XF_{vj} \leqslant 1,\ \forall\ j \in J_d \tag{43}$$

$$\sum_{j \in J_l} XF_{ij} + \sum_{i' \in I: i' \neq i} X_{i'il} + \sum_{v \in V} X_{vil} = 1,\ \forall\ i \in I,\ l \in L_d \tag{44}$$

$$\sum_{j \in J_l} XF_{vj} + \sum_{i \in I} X_{ivl} = \sum_{j \in J_l} Y_{vj},\ \forall\ v \in V,\ l \in L_d \tag{45}$$

$$\sum_{i' \in I: i' \neq i} X_{ii'l} + \sum_{v \in V} X_{ivl} + \sum_{j \in J_l} XL_{ij} = 1,\ \forall\ i \in I,\ l \in L_d \tag{46}$$

$$\sum_{i \in I} X_{vil} \leqslant \sum_{j \in J_l} Y_{vj},\ \forall\ v \in V,\ l \in L_d \tag{47}$$

$$Y_{ij} \leqslant Y_{vj} + 1 - X_{ivl} - X_{vil},\ \forall\ i \in I,\ v \in V,\ j \in J_l, l \in L_d \tag{48}$$

$$Y_{vj} \leqslant Y_{ij} + 1 - X_{vil} - X_{ivl},\ \forall\ i \in I,\ v \in V,\ j \in J_l, l \in L_d \tag{49}$$

Because of resource limitation, maintenance is not available during certain periods, such as weekends and holidays. These break periods for maintenance are denoted by special orders $q \in Q$ with fixed time parameters te_{ql} and ts_{ql}. New sequencing binary variables X_{vql}^{g} are introduced to indicate whether maintenance order v precedes order q in any unit of stage $l \in L_d$ or not. The logic constraints for the timing of order q and v are obtained accordingly as Eq. (50) illustrates, in which maintenance order v is either before or after order q in stage l. The reformulated big-M constraints are presented in Eq. (51).

$$\begin{bmatrix} X^g_{vql} \\ Ts_{vl} \geqslant te_{ql} \end{bmatrix} \veebar \begin{bmatrix} \neg X^g_{vql} \\ Te_{vl} \leqslant ts_{ql} \end{bmatrix}, \ \forall \ v \in V, \ q \in Q, \ l \in L_d \tag{50}$$

$$\begin{cases} Ts_{vl} \geqslant te_{ql} - M \cdot (1 - X^g_{vql}), \\ Te_{vl} \leqslant ts_{ql} + M \cdot X^g_{vql}, \end{cases} \forall \ v \in V, \ q \in Q, \ l \in L_d \tag{51}$$

3.3.2. C2: maintenance operations as an option before each production order

Instead of treating maintenance operations as one type of order, maintenance is taken as an option before each production order in formulation $C2$. Let Z_{il} denote a maintenance order to be undertaken right before order i in stage l. As a result, new constraints are developed regarding the fouling evolution and the timing of any two orders. The logical constraints regarding fouling evolution in Eq. (21) are extended as Eq. (52) demonstrates, where Z_{il} is taken in representing the disjunctions that consider maintenance. For example, $X_{i'il} \wedge Y_{i'j} \wedge Y_{ij} \wedge (\neg Z_{il}) = 1$ means that order i is sequenced immediately after order i' with no maintenance processed right before order i. The corresponding big-M reformulated constraints are presented in Eqs. (53)–(55).

$$\underset{i' \in I: i \neq i'}{\veebar} \begin{bmatrix} X_{i'il} \wedge Y_{i'j} \wedge Y_{ij} \wedge (\neg Z_{il}) \\ f_{ij} = AS_{ij} \cdot f_{i'j} + BS_{ij} \end{bmatrix} \veebar \begin{bmatrix} XF_{ij} \wedge (\neg Z_{il}) \\ f_{ij} = FI_j \end{bmatrix} \veebar \begin{bmatrix} Y_{ij} \wedge Z_{il} \\ f_{ij} = F_c \end{bmatrix} \veebar \begin{bmatrix} \neg Y_{ij} \\ f_{ij} = 0 \end{bmatrix}, \quad \forall \ i \in I, \ j \in J_l, \ l \in L_d \tag{52}$$

$$\begin{cases} f_{ij} \geqslant AS_{i'j} \cdot f_{i'j} + BS_{i'j} - M^{26}_{i'ij} \cdot (3 - X_{i'il} - Y_{i'j} - Y_{ij} + Z_{il}), \\ f_{ij} \leqslant AS_{i'j} \cdot f_{i'j} + BS_{i'j} + M^{27}_{i'ij} \cdot (3 - X_{i'il} - Y_{i'j} - Y_{ij} + Z_{il}), \end{cases}$$
$$\forall \ i, i' \in I : i' \neq i, \ j \in J_l, \ l \in L_d \tag{53}$$

$$\begin{cases} f_{ij} \geqslant FI_j - M^{28}_{ij} \cdot (1 - XF_{ij} + Z_{il}), \\ f_{ij} \leqslant FI_j + M^{29}_{ij} \cdot (1 - XF_{ij} + Z_{il}), \end{cases} \forall \ i \in I, \ j \in J_l \ l \in L_d \tag{54}$$

$$\begin{cases} f_{ij} \geqslant F_c - M^{40}_{ij} \cdot (2 - Y_{ij} - Z_{il}), \\ f_{ij} \leqslant F_c + M^{41}_{ij} \cdot (2 - Y_{ij} - Z_{il}), \end{cases} \quad \forall \ i \in I, \ j \in J_l, \ l \in L_d \tag{55}$$

Similarly, the logic constraints in Eq. (8) is extended for the timing of orders as Eq. (56) shows, where $tm \cdot Z_{il}$ is added to account for maintenance. The corresponding big-M constraints are shown in Eq. (57).

$$\underset{i,i' \in I: i \neq i', \ l \in L_d}{\veebar} \begin{bmatrix} X_{i'il} \\ Te_{i'l} + Tw_{i'l} + tr_{i'l} + tm \cdot Z_{il} \leqslant Ts_{il} \end{bmatrix} \tag{56}$$

$$Te_{i'l} + Tw_{i'l} + tr_{i'l} + tm \cdot Z_{il} \leqslant Ts_{il} + M \cdot (1 - X_{i'il}),$$
$$\forall \ i, i' \in I : i \neq i', \ l \in L_d \tag{57}$$

Considering break periods $q \in Q$ in formulation $C2$, new sequencing variables X^g_{iql} along with Z_{il} are employed to indicate whether a maintenance operation exists right before order i and also precedes q or not. As a result, the logic constraints in Eq. (59) only allow maintenance operations to be undertaken before or after break period q, where the corresponding big-M constraints are presented in Eq. (60).

$$X^g_{iql} + X^g_{qil} = Z_{il}, \ \forall i \in I, \ q \in Q, \ l \in L_d \tag{58}$$

$$\begin{bmatrix} X^g_{iql} \\ Ts_{il} \leqslant ts_q \end{bmatrix} \veebar \begin{bmatrix} X^g_{qil} \\ te_q \leqslant Ts_{il} - tm \end{bmatrix}, \ \forall \ i \in I, \ q \in Q, \ l \in L_d \tag{59}$$

$$\begin{cases} Ts_{il} \leqslant ts_q + M \cdot (1 - X^g_{iql}), \\ te_q \leqslant Ts_{il} - tm + M \cdot (1 - X^g_{qil}), \end{cases} \forall \ i \in I, \ q \in Q, \ l \in L_d \tag{60}$$

3.4. Objective function

Minimization of makespan (MS) or the end time of the last order in the production sequence is considered as the objective function in the proposed scheduling formulation. MS as defined in Eq. (62) is a metric to indicate production efficiency in batch scheduling.

$$\min \ MS \tag{61}$$

$$MS \geqslant Te_{il}, \ \forall \ i \in I, \ l = |L| \tag{62}$$

The evolution of fouling can also be integrated in the objective function when maintenance is considered in the formulation. Maintenance restores degradation and leads to reduced values of the fouling KPI, and lower fouling values result in lower batch processing time. On the other hand, the duration of maintenance contributes negatively to the minimization of MS. Given the KPI threshold as a hard constraint to prevent the failure of units, each unit may have non-production time during the scheduling due to a larger-than-threshold value of the KPI. When minimizing the MS maintenance is only taken when the non-production time in one unit is larger than the time cost of maintenance. On the other hand, the final fouling KPI, namely the fouling KPI of the scheduled last order in each unit, declines by performing maintenance and will become a smaller initial KPI in the short-term scheduling problem for the next time horizon, which means a longer remaining useful life for the degraded units. When minimizing only MS, the contribution of maintenance to future production capacity is not considered in the scheduling. As a result, a new objective function incorporates penalty terms of final KPIs in each of the degraded units and is presented in Eq. (63).

$$\min \ \lambda MS + (1 - \lambda) \sum_{j \in J_d} G_j(f_{fi,j}) \tag{63}$$

where, $f_{fi,j}$ denotes the final fouling KPI at unit j; $\sum_{j \in J_d} G_j(f_{fi,j})$ represents the time-related measurement for the fouling in the units, which leads to maximum remaining useful life by minimizing the measurement; the time-related measurement is the reverse of the percentage value for the units' remaining useful life. The evolution of the fouling KPI in the example displays nonlinear behavior, this means that the KPI evolution rate increases more drastically towards the end of a campaign than at the beginning. As a result, batch runs that represent the life time of units, contribute differently to the fouling evolution, depending on its position in the sequence of a campaign. Additionally the fouling value is not directly proportional to the life time of unit; for example, the unit's life time with respect to fouling reaches 80% when the value of the KPI reaches 40%. To mitigate the nonlinearity, $G_j(\ \cdot\ )$ gives the time-related fouling measurement as a possible nonlinear function of the fouling KPI.

The final fouling KPI is not always the largest value in the series of fouling KPI due to maintenance leading to difficulties in finding the representation of $f_{fi,j}$. One method is to use GDP-based constraints in accordance to XL_{ij}, which denotes the last order of each unit and therefore finds the corresponding KPI. Another method is to introduce auxiliary orders $i \in I^{aux}_j$ that are assigned to the last place of each units as illustrated in Eq. (64). As a result, final KPI

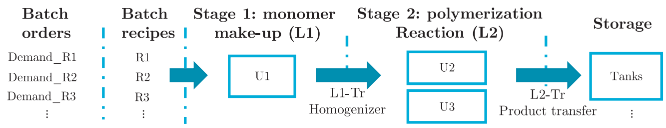

Fig. 6. The topology of the batch process

values in different units are obtained from the fouling KPI of the auxiliary orders as Eq. (65) shows.

$$XL_{ij} = 1, \ \forall \ i \in I_j^{aux}, \ \ j \in J_d \tag{64}$$

$$f_{fi,j} = f_{i,j}, \ \forall \ i \in I_j^{aux}, \ \ j \in J_d \tag{65}$$

4. Case study and computational results

In this section, a detailed version of the two-stage multiproduct batch process is presented. The scheduling formulations outlined in Section 3 are then applied to this specific scenario and compared based on their computational performance.

The production procedures of the case study plant consist of monomer make-up, homogenisation, and reaction. Raw materials are added to the monomer make-up vessel in a specific order and circulated around a heat exchanger to cool. The prepared monomer is then fed to the homogenisers where it is mixed in a specific ratio with oil. In the homogenisers, a monomer emulsion is formed and directly transferred into a batch reactor. After charging the emulsion, polymerization begins with the inflow of initiators. To ensure temperature control of the reactor, the emulsion is circulated through external heat exchangers. Once finished, the product is transferred to one of many storage tanks via a route selection manifold. The batch production follows recipes, which specify the charge amount of raw materials, reaction profiles, the size of batch run, etc, which determines the product grade for each batch run.

In accordance with the case study, the two-stage batch process has one monomer make-up vessel U1 in stage L1 and two batch reactors U2 and U3 in stage L2 as Fig. 6 shows. The material transfer L1-Tr is between the two stages and occurs via the homogenizers, which in turn charge the reactors with the reactants. The prepared product is transferred from reactors to further storage tanks through L2-Tr. Multiple orders of several different recipes are dispatched to the process to produce multiple grades of chemicals.

As previously mentioned, precedence-based models are employed to represent the aforementioned scheduling problem. In the two stages of the process, stage L1 has only one unit; stage L2 has two units, both of which have sequence-dependent degradation. An immediate precedence-based system is employed for the scheduling in stage L2 to handle sequence-dependent fouling, while both precedence models can be applied to schedule in stage L1. This leads to two scheduling formulations M1 and M2 which were originally described in Wu et al. (2019b). These will be compared to the models where symmetry-breaking constraints are considered, henceforth known as M1S and M2S respectively. Furthermore, if restorative maintenance is considered in stage L2 the formulation names are appended with C1, and C2, representing the original formulation as well as one of the two proposed approaches to model the maintenance actions. The scheduling formulations are summarized in Table 1.

The formulations in Table 1 are tested with two different objective functions. The first one is minimizing the MS. For formulations that consider both scheduling and maintenance, an alternative objective that minimizes both MS and final KPI values is presented in Eq. (63). In this case, $G_j(\ \cdot\ )$ interprets the final fouling KPI as the time-related index to show the unit's life that has been used with respect to fouling. Minimizing the function $G_j(\ \cdot\ )$ is essentially maximization of the remaining number of batches that can be run before next maintenance. The evolution rate for the fouling KPI rate varies over the course of a production campaign. As can be seen from the sequence-dependent model in Eq. (1), the larger the fouling KPI value becomes, the faster the rate of future increase. As a result, a piecewise linear function is taken to approximate the curved shape of the batch-to-batch fouling evolution (see Eq. (66)). It partitions the range of the KPI into three pieces that are linear between the breakpoints P_1 and P_2. The slope parameters of the three pieces are proportional to the average evolution rate of the fouling KPI in each segment. The linear segment with the lowest KPI value corresponds to the largest value of C_1^p as more batches can be produced early in the campaign than can be produced later. A set of binary variables $\{X_{jn}^{pw}\}$ denotes whether the final KPI values are larger or smaller than each breakpoint P_n. The continuous variables $\{F_{j1}, F_{j2}, F_{j3}\}$ indicate the weights of the final fouling KPI value in each segment respectively and are obtained from the GDP model using the breakpoint binary variables as presented in Eqs. (67) and (68).

$$G_j = \sum_{n=1}^{3} C_n^p F_{jn} + C_1^p (P_1 - Fc) X_{j1}^{pw} + C_2^p (P_2 - P_1) X_{j2}^{pw}, \ \forall \ j \in J_d \tag{66}$$

$$\begin{bmatrix} \neg X_{jn}^{pw} \\ f_{fi,j} < P_n \end{bmatrix} \underline{\bigvee} \begin{bmatrix} X_{jn}^{pw} \\ f_{fi,j} \geqslant P_n \end{bmatrix}, \ \forall \ j \in J_d, \ \ n \in \{1, 2\} \tag{67}$$

$$\begin{bmatrix} \neg X_{j1}^{pw} \\ F_{j1} \geqslant f_{fi,j} - Fc \end{bmatrix} \underline{\bigvee} \begin{bmatrix} X_{j1}^{pw} \wedge \neg X_{j2}^{pw} \\ F_{j2} \geqslant f_{fi,j} - P_1 \end{bmatrix} \underline{\bigvee} \begin{bmatrix} X_{j2}^{pw} \\ F_{j3} \geqslant f_{fi,j} - P_2 \end{bmatrix}, \ \forall \ j \in J_d \tag{68}$$

4.1. Computational results

In this section, the proposed scheduling formulations for the given case study problems are tested for a variety of different problem sizes and features. The size of the short-term scheduling problem considers at most 20 orders and up to three unique recipes. The scheduling models were coded in GAMS v26.1. The computational results were obtained on a Windows 10 computer with an Intel i7 (4.2 GHz and 4 cores) processor and 32GB of RAM using CPLEX 12.8 with four threads and maximum computational time 3600 s to solve the resulting MILPs.

Scheduling without maintenance The scheduling formulations considering no maintenance are compared by testing them with different problem sizes. Scheduling problem instances CS1 to CS7 are taken from the case study example (see Fig. 6) with two stages and three units; in the second stages, two reactors have sequence-dependent fouling evolution (see Eq. (1)) and have fouling-dependent batch duration (see Eq. (2)). Symbols $|I|$ represents total numbers of orders and the actual order numbers of each recipe added in parentheses as Table 2 shows, while $|R|$ represents numbers of recipes in each problem, which together determines the problem size. A list of parameters for the problem instances is presented in Nomenclature, which shows the features of operations in the industrial examples such as timing parameters and degradation-related measurement. The objective of the scheduling algorithm is to minimize the MS, and the corresponding computational results are summarized in Tables 2 and 3. Optimal solutions for the scheduling problem using different formulations (see Table 1) are obtained by running optimization with larger time limitation and presented in Table 3. However, problems with relatively larger size such as CS4 and CS6, still cannot be solved to the optimal solution within a few days and are presented with missing optimal objective values.

Table 1
Model descriptions: ImP denotes immediate precedence model, GeP denotes general precedence model.

Models	M1	M2	M1S	M2S	M2SC1	M2SC2
ImP for both stage L1 and L2	✓		✓			
GeP for stage L1, ImP for stage L2		✓		✓	✓	✓
Symmetry-breaking constraints			✓	✓	✓	✓
Maintenance using formulation C1					✓	
Maintenance using formulation C2						✓

Table 2
Computational effort (CPU time (sec) & relative optimality gap) for production scheduling without maintenance: the best performer is in bold.

Prob.	$\|I\|$	$\|R\|$	M1	M2	M1S	M2S
CS1	12(6,6)	2	90(0%)	3600(26%)	21(0%)	**5(0%)**
CS2	12(4,4,4)	3	1300(0%)	3600(24%)	584(0%)	**262(0%)**
CS3	15(8,7)	2	1542(0%)	3600(31%)	1946(0%)	**150(0%)**
CS4	15(5,5,5)	3	3600(46%)	3600(31%)	3600(44%)	3600(27%)
CS5	18(9,9)	2	3600(50%)	3600(35%)	3600(10%)	3600(6%)
CS6	18(6,6,6)	3	3600(60%)	3600(34%)	3600(52%)	3600(30%)
CS7	20(10,10)	2	3600(57%)	3600(38%)	3600(40%)	3600(18%)

Table 3
Solutions with objective values (MS) for production scheduling without maintenance.

Prob.	$\|I\|$	$\|R\|$	Opt.	M1	M2	M1S	M2S
CS1	12(6,6)	2	5069	5069	5069	5069	5069
CS2	12(4,4,4)	3	5018	5018	5018	5018	5018
CS3	15(8,7)	2	5898	5898	5898	5898	5898
CS4	15(5,5,5)	3	–	5840	5841	5840	5841
CS5	18(9,9)	2	6741	6745	6744	6743	6741
CS6	18(6,6,6)	3	–	6672	6669	6672	6662
CS7	20(10,10)	2	7291	7293	7298	7292	7291

By adding symmetry-breaking constraints, M1S and M2S show a noticeable improvement in regards to computational efficiency when compared with M1 and M2. M1S and M2S use less CPU time to solve problems CS1, CS2 and CS3 to global optimality, as shown in Table 2. As the problem size becomes larger, the solutions found by M1S and M2S within one hour are also superior to the ones of M1 and M2 as the relative optimality gaps in Table 2 and the objective function values in Table 3 indicate. The general precedence model has computational advantages over the immediate precedence model as the former has fewer binary variables and simplifies the model, which is increasingly evident as the problem size grows. Comparing with M1, M2 uses the general precedence model in stage L1 instead of the immediate precedence model and shows better performance for the larger problem sizes. On the other hand M1 shows superior performance over M2 in smaller-size problems (CS1, CS2 and CS3) taking less time to find the provably optimal solutions. By including symmetry breaking, the new version of M2 performs better than M1 demonstrating lower CPU time or smaller objective values for all the problems in Tables 2 and 3. This is because the symmetry breaking based the general precedence-based structure in Eq. (32) adds tighter constraints to the formulation than the one for the immediate precedence-based structure in Eq. (33). As expected, more types of recipes in the scheduling reduces the effectiveness of the symmetry-breaking constraints, because, as the number of recipe increases, the problem with the same number of orders has fewer number of equivalent solutions that can be ruled out with the said constraints. When the problem size reaches 15 orders and three recipes, even model M2S cannot prove optimality within a one-hour computation time limit. On the other hand, the optimal solutions are obtained for CS5 and CS7 by running optimization for much longer time (several days), but the ones for CS4 and CS6 are still missing due to large computational cost. For the corresponding incumbent solutions of CS5 and CS7 at one-hour time limit, the actual optimality gaps turn out to be zero even if the relative optimality gaps are around 6% and 18%.

The Gantt chart and the fouling evolution curves for the M2S solution to problem CS6 are presented in Fig. 7. In the figure, all units are available from day one, and R1, R2, and R3 denote the recipes of each order using different colors or symbols. The normalized fouling KPI (with a max value of one) shows the evolution of fouling in the two units given the planned production sequence. To provide a point of comparison, one rule-based scheduling heuristic (which is often used in practice) is also applied to the case study. In the example, a specific dispatching rule assigns orders to the units following a pre-defined sequence of recipes. Firstly, the heuristic finishes orders of recipe R2, then recipe R1, and afterwards recipe R3. The dispatching rule is interpreted as heuristic constraints, which simulates how manual scheduling is done in practice by adding the heuristic sequencing constraints to the scheduling formulation and running the reduced optimization problem. The solution of the rule-based heuristic for 18 orders and three recipes is illustrated by the Gantt chart and fouling evolution curves in Fig. 8. The rule-based heuristic generates suboptimal solutions with an objective value of 6712 min, larger than the M2S solution of 6662 min. While the difference in makespan admittedly is not very large, we will see that the scheduling formulation has further advantages, in terms of including maintenance and breaking periods in the scheduling

Scheduling considering maintenance Considering maintenance in the scheduling problem, formulation M2S is taken (as it showed the best computational performance) and integrated with the two aforementioned maintenance formulations C1 and C2. Seven problem instances CM1 to CM7 are created based on instances CS1 to CS2. These new instances additionally consider maintenance scheduling and its corresponding unavailability periods in units of the second stage. The scheduling problems are solved with the goal of minimizing the MS using the two different formulations. The resulting computational results are presented in Table 4. Comparing the problem sizes for CM6, the reduced MIP of formulation C1 has 4184 rows, 562 columns (470 binaries), and 15042 nonzeros, while formulation C2 has 3770 rows, 628 columns (540 binaries), and 15,006 nonzeros. It shows that formulation C1 has a reduced number of binaries to deal with maintenance periods compared to C2 but has more constraints to couple the maintenance tasks to production. The two formulations have different strengths in han-

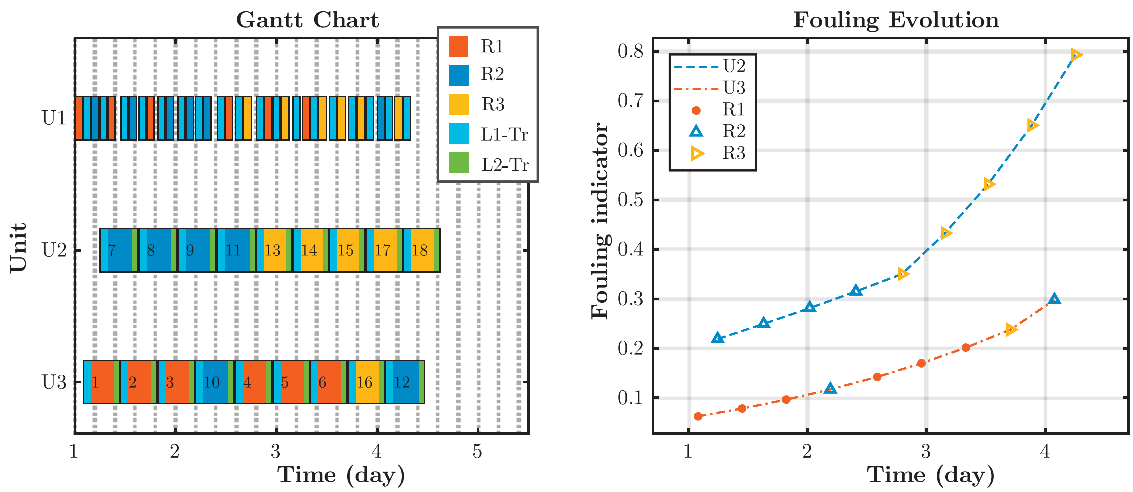

Fig. 7. Gantt chart and fouling evolution: M2S solution for 18 orders and three recipes.

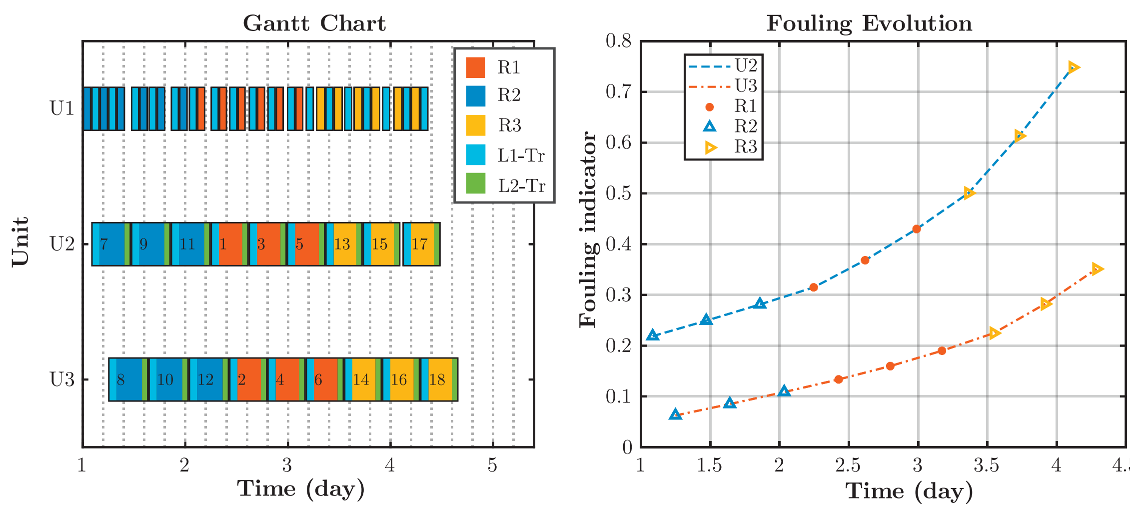

Fig. 8. Gantt chart and fouling evolution: a rule-based schedule for 18 orders and three recipes.

Table 4
Computational results for production scheduling (M2S) with maintenance.

Prob.	|I|	|R|	Opt.	CPU time (relative optimality gap)		Obj. MS	
				C1	C2	C1	C2
CM1	12(6,6)	2	6420	89(0%)	52(0%)	6420	6420
CM2	12(4,4,4)	3	6414	3600(30%)	336(0%)	6414	6414
CM3	15(8,7)	2	7393	3600(19%)	3547(0%)	7393	7393
CM4	15(5,5,5)	3	–	3600(43%)	3600(41%)	7413	7413
CM5	18(9,9)	2	–	3600(19%)	3600(16%)	8191	8191
CM6	18(6,6,6)	3	–	3600(47%)	3600(46%)	8471	8436
CM7	20(10,10)	2	–	3600(36%)	3600(25%)	8933	8808

dling different features of scheduling problems. In this example, the result in Table 4 show that formulation M2SC2 performs better than M2SC1, as M2SC2 solved problems CM1, CM2, and CM3 to optimality with less CPU time and obtained better solutions and smaller relative optimality gaps than M2SC1 in the problems with larger-than-15 orders.

Breaking periods are another thing to consider. Maintenance cannot be performed during breaking periods. In certain situation this leads to the unavailability of a unit for a long time as the unit cannot be run due to its fouling level, but also cannot be maintained due to break periods. One scenario explaining this is shown in Fig. 9. In this example, the dispatching rule dictates that maintenance should be taken when the fouling level of a unit reaches 80% of the threshold. The units are available to produce since the noon of Wednesday for this short-term schedule, and the initial normalized fouling KPIs are 0.68 and 0.31 for the two reaction units respectively. Weekends are considered as the breaking periods during the scheduling horizon. The simulated manual scheduling does not foresee the demand for maintenance and therefore misses the timing of maintenance before the breaking period (denoted by the greyed out area) and therefore ends up with a large value for MS. The same example using the proposed formulation takes these features into account, and it generates the solution illustrated in Fig. 10. The maintenance is done before the breaking period allowing unit U2 to produce during the breaking period, which saves a considerable amount of production time (3947 min).

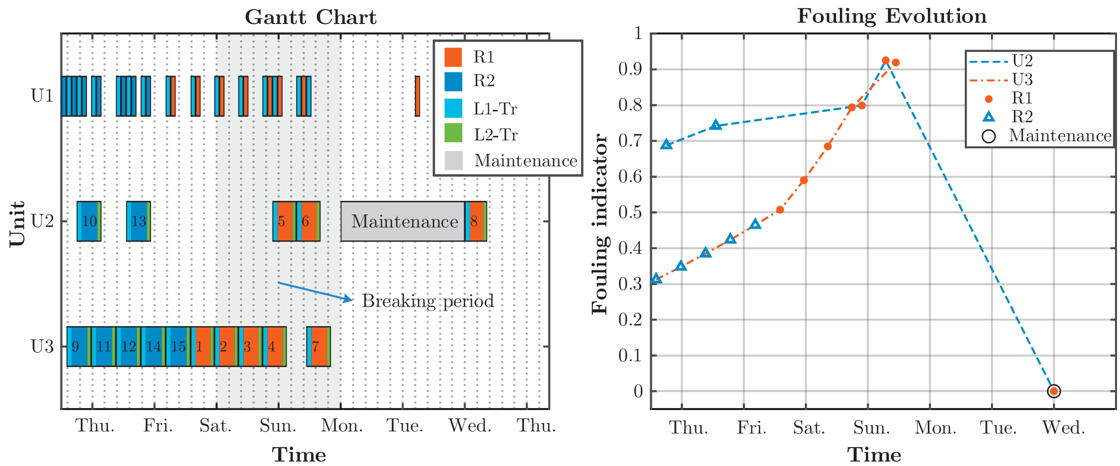

Fig. 9. Gantt chart and fouling evolution: a rule-based scheduling for 15 orders and two recipes considering maintenance.

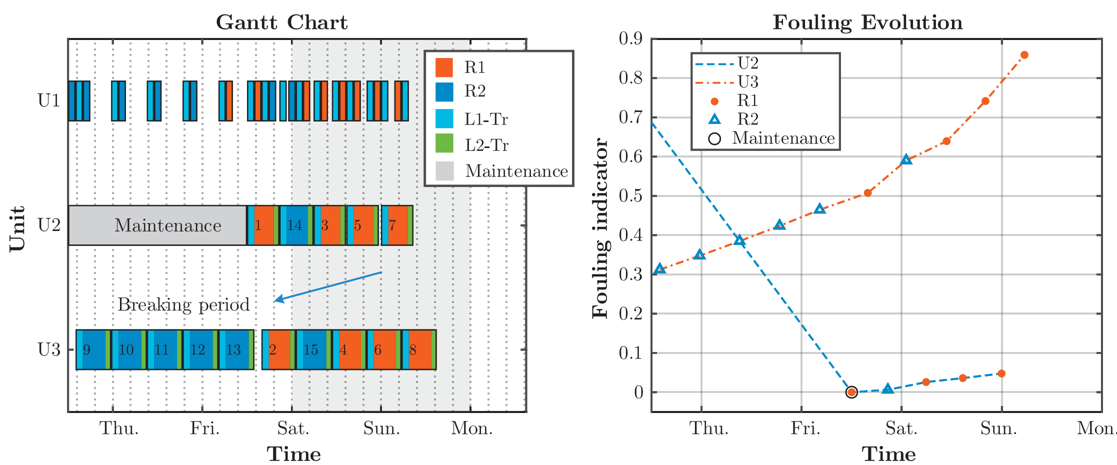

Fig. 10. Gantt chart and fouling evolution: M2SC2 for 15 orders and two recipes considering maintenance.

Multiple objectives There are often competing objectives when considering maintenance in scheduling. On the one hand, maintenance is necessary to restore a unit from a degraded state to a healthy state. On the other hand, maintenance costs time, which negatively affects the throughput of a plant. Therefore, there is a trade-off between finishing the current set of orders in a timely manner and ensuring the production units in sufficient condition to operate when future orders arrive. To investigate this trade-off a Pareto front is generated for a set of scheduling problems (Mo1, Mo2 and Mo3) based on M2SC2 with different initial fouling KPIs, in which the weight parameter (λ) balances the two objectives and takes series of numbers in the range from zero to one. Regarding the piecewise function of the final KPI level G_j in Eq. (66), the ratio of the slope parameters ($C_1^p : C_2^p : C_3^p$) of the piecewise weight variables is 10: 7: 4. Each scheduling problem generates up to three points representing different solutions as Fig. 11 demonstrates, in which many repeated solutions are generated with different values of λ. This is because of the discrete characteristics of the problem that the objective term regarding the remaining useful life is mainly determined by times of maintenance while minimizing the MS. In the horizon for short-term scheduling, each unit that has performance decay allows maintenance to be performed at most one time leading to up to three types of solutions. These three solutions are: no maintenance in any reaction units, maintenance in the reaction unit that has higher fouling KPI, and maintenance in both units. In some cases, the solution for minimizing MS only decides to have maintenance in one reaction unit as Fig. 10 shows ending up with only two types of solutions in the Pareto front.

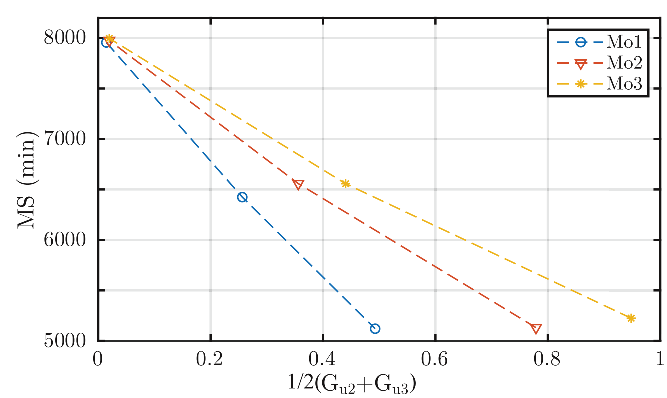

Fig. 11. Pareto front illustrating the trade-off between MS and final KPI level.

The Pareto front shows that the more emphasis that is placed on the unit-average final time-related fouling index $(G_{u2} + G_{u3})/2$, logically results in a larger MS because carrying out maintenance costs additional time. A qualitative analysis of the results is presented. The Gantt chart and fouling evolution curves for minimizing MS are compared with the ones for scheduling with large emphasis on maximizing residual useful life. The Gantt chart and fouling evolution curve shown in Fig. 12 comes from problem Mo3 in Fig. 11. In this case, no maintenance is performed since the fouling KPI does not reach the threshold though it comes very close. By

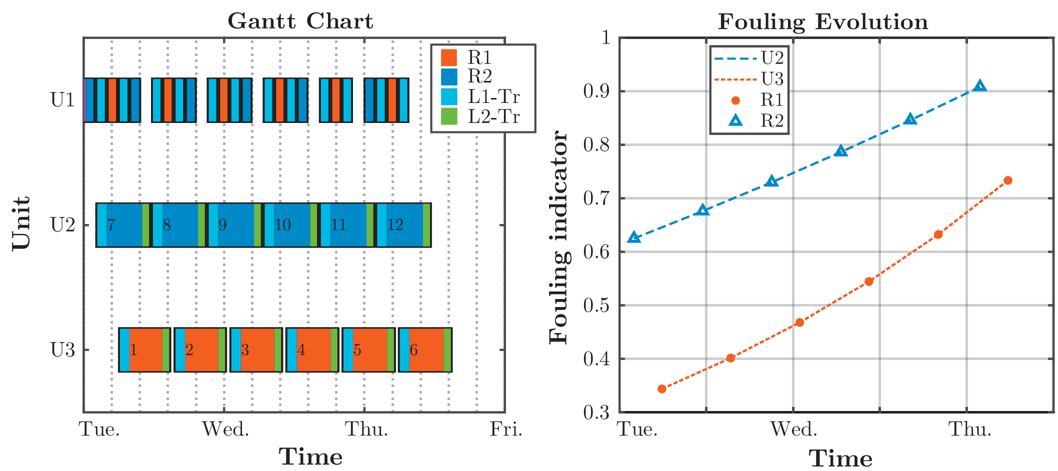

Fig. 12. Gantt chart and fouling evolution for scheduling that minimizes MS.

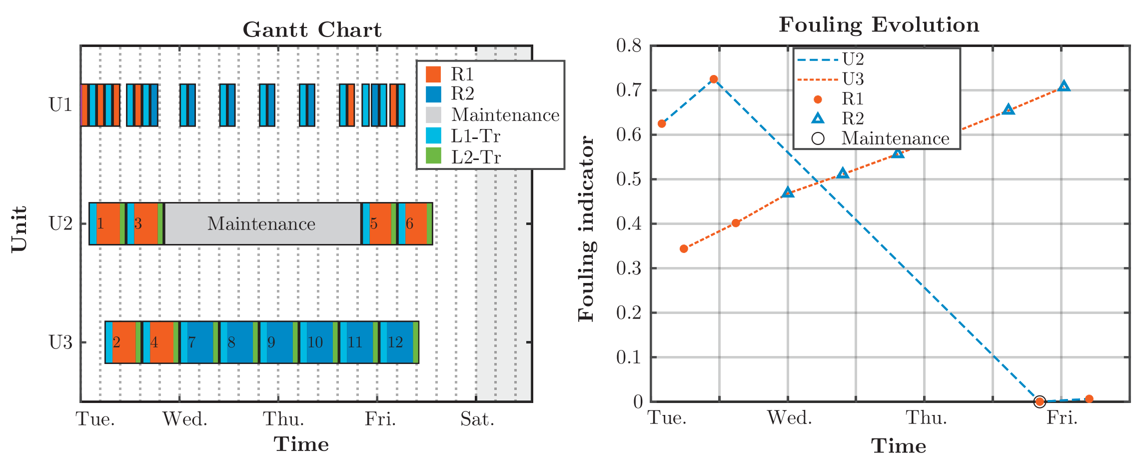

Fig. 13. Gantt chart and fouling evolution for scheduling with a large emphasis on residual useful life.

placing more weight on the final KPI level for problem M3, the result (presented in Fig. 13) shows that maintenance is performed at unit U2, and the normalized final fouling KPI in U2 is restored to around zero. By maximizing residual useful life or minimizing final fouling level maintenance is arranged earlier than it has to be, which is desirable as production in practice is performed in a continuous manner, and the health of the unit at the end of the current scheduling horizon will greatly impact the schedule of the upcoming horizons. To find an appropriate weight parameter to strike a balance between short-term scheduling and the cost of maintenance in the long term as well as further multiobjective optimization methods are left as future work.

5. Conclusions

In this work, a short-term scheduling model for multiproduct batch processes is proposed, which considers sequence-dependent degradation and associated maintenance actions. The continuous-time MILP approach integrates degradation models into the precedence-based formulations using GDP and big-M reformulation methods to account for performance decay.

A polymerization case study is outlined and solved using the proposed scheduling formulations. The results illustrate the computational performance of the short-term scheduling showing that novel symmetry-breaking constraints enhance the optimization efficiency. Furthermore, two types of maintenance formulations are compared and analyzed to show the computational features of each one. The proposed scheduling solutions are further compared with simulated manual scheduling to showcase the potential benefits of the proposed approach. Moreover, the analysis is extended to multiobjective optimization to characterize the trade-off between speeding production in the current scheduling horizon and having sufficiently healthy equipment to produce in future time periods. The bi-objective problems are visualized to show the trade-off between the two.

Future work could investigate further decomposition-based approaches to solve efficiently such scheduling models. Additionally, the tradeoffs between short-term scheduling, and longer-term maintenance planning could also be an interesting point of future investigation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Ouyang Wu: Conceptualization, Methodology, Software, Investigation, Writing - original draft, Visualization. **Giancarlo Dalle Ave:** Methodology, Software, Writing - review & editing. **Iiro Harjunkoski:** Conceptualization, Methodology, Writing - review & editing. **Ala Bouaswaig:** Writing - review & editing, Resources. **Stefan Marco Schneider:** Writing - review & editing, Supervision.

Matthias Roth: Writing - review & editing, Resources. **Lars Imsland:** Writing - review & editing, Supervision.

Acknowledgements

Financial support is gratefully acknowledged from the Marie Skodowska Curie Horizon 2020 EID-ITN project "PROcess NeTwork Optimization for efficient and sustainable operation of Europe's process industries taking machinery condition and process performance into account PRONTO", Grant agreement no 675215.

References

Aguirre, A.M., Papageorgiou, L.G., 2018. Medium-term optimization-based approach for the integration of production planning, scheduling and maintenance. Comput. Chem. Eng. 116, 191–211.
Baumann, P., Trautmann, N., 2014. A hybrid method for large-scale short-term scheduling of make-and-pack production processes. Eur. J. Oper. Res. 236 (2), 718–735.
Biondi, M., Sand, G., Harjunkoski, I., 2017. Optimization of multipurpose process plant operations: a multi-time-scale maintenance and production scheduling approach. Comput. Chem. Eng. 99, 325–339.
Bott, T.R., 1995. Fouling of Heat Exchangers. Elsevier.
Castro, P.M., Grossmann, I.E., 2012. Generalized disjunctive programming as a systematic modeling framework to derive scheduling formulations. Ind. Eng. Chem. Res. 51 (16), 5781–5792.
Castro, P.M., Grossmann, I.E., Veldhuizen, P., Esplin, D., 2014. Optimal maintenance scheduling of a gas engine power plant using generalized disjunctive programming. AIChE J. 60 (6), 2083–2097.
Cerdá, J., Henning, G.P., Grossmann, I.E., 1997. A mixed-integer linear programming model for short-term scheduling of single-stage multiproduct batch plants with parallel lines. Ind. Eng. Chem. Res. 36 (5), 1695–1707.
Dalle Ave, G., Hernandez, J., Harjunkoski, I., Onofri, L., Engell, S., 2019. Demand side management scheduling formulation for a steel plant considering electrode degradation. IFAC-PapersOnLine 52 (1), 691–696.
Dedopoulos, I.T., Shah, N., 1995. Optimal short-term scheduling of maintenance and production for multipurpose plants. Industrial & engineering chemistry research 34 (1), 192–201.
Erdirik-Dogan, M., Grossmann, I.E., 2008. Slot-based formulation for the short-term scheduling of multistage, multiproduct batch plants with sequence-dependent changeovers. Ind. Eng. Chem. Res. 47 (4), 1159–1183.
Gorjian, N., Ma, L., Mittinty, M., Yarlagadda, P., Sun, Y., 2010. A review on degradation models in reliability analysis. In: Engineering Asset Lifecycle Management. Springer, pp. 369–384.
Gupta, S., Karimi, I., 2003. An improved milp formulation for scheduling multiproduct, multistage batch plants. Ind. Eng. Chem. Res. 42 (11), 2365–2380.
Harjunkoski, I., Maravelias, C.T., Bongers, P., Castro, P.M., Engell, S., Grossmann, I.E., Hooker, J., Méndez, C., Sand, G., Wassick, J., 2014. Scope for industrial applications of production scheduling models and solution methods. Comput. Chem. Eng. 62, 161–193.
Margot, F., 2010. Symmetry in integer linear programming. In: 50 Years of Integer Programming 1958–2008. Springer, pp. 647–686.
Martin, P., Strutt, J., Kinkead, N., 1983. A review of mechanical reliability modelling in relation to failure mechanisms. Reliab. Eng. 6 (1), 13–42.
Méndez, C., Henning, G., Cerdá, J., 2000. Optimal scheduling of batch plants satisfying multiple product orders with different due-dates. Comput. Chem. Eng. 24 (9–10), 2223–2245.
Méndez, C., Henning, G., Cerdá, J., 2001. An milp continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. Comput. Chem. Eng. 25 (4–6), 701–711.
Méndez, C.A., Cerdá, J., 2003. An milp continuous-time framework for short-term scheduling of multipurpose batch processes under different operation strategies. Optim. Eng. 4 (1–2), 7–22.
Méndez, C.A., Cerdá, J., Grossmann, I.E., Harjunkoski, I., Fahl, M., 2006. State-of-the-art review of optimization methods for short-term scheduling of batch processes. Comput. Chem. Eng. 30 (6–7), 913–946.
Teruel, E., Cortes, C., Diez, L.I., Arauzo, I., 2005. Monitoring and prediction of fouling in coal-fired utility boilers using neural networks. Chem. Eng. Sci. 60 (18), 5035–5048.
Vieira, M., Pinto-Varela, T., Barbosa-Póvoa, A.P., 2017. Production and maintenance planning optimisation in biopharmaceutical processes under performance decay using a continuous-time formulation: a multi-objective approach. Comput. Chem. Eng. 107, 111–139.
Wu, O., Bouaswaig, A., Imsland, L., Schneider, S.M., Roth, M., Leira, F.M., 2019. Campaign-based modeling for degradation evolution in batch processes using a multiway partial least squares approach. Comput. Chem. Eng. 128, 117–127.
Wu, O., Dalle Ave, G., Harjunkoski, I., Imsland, L., Schneider, S.M., Bouaswaig, A.E.F., Roth, M., 2019. Short-term scheduling of a multipurpose batch plant considering degradation effects. In: Computer Aided Chemical Engineering, 46. Elsevier, pp. 1213–1218.
Wu, O., Imsland, L., Brekke, E., Schneider, S.M., Bouaswaig, A.E., Roth, M., 2019. Robust state estimation for fouling evolution in batch processes using the em algorithm. IFAC-PapersOnLine 52 (1), 685–690.
Xenos, D.P., Kopanos, G.M., Cicciotti, M., Thornhill, N.F., 2016. Operational optimization of networks of compressors considering condition-based maintenance. Comput. Chem. Eng. 84, 117–131.
Yeap, B., Wilson, D., Polley, G., Pugh, S., 2004. Mitigation of crude oil refinery heat exchanger fouling through retrofits based on thermo-hydraulic fouling models. Chem. Eng. Res. Des. 82 (1), 53–71.
Zhang, J., Morris, A., Martin, E., Kiparissides, C., 1999. Estimation of impurity and fouling in batch polymerisation reactors through the application of neural networks. Comput. Chem. Eng. 23 (3), 301–314.
Zmitrowicz, A., 2006. Wear patterns and laws of wear–a review. J. Theor. Appl. Mech. 44 (2), 219–253.