Integrated Scheduling and Dynamic Optimization of Sequential Batch Processes with Online Implementation

Yunfei Chu and Fengqi You
Dept. of Chemical and Biological Engineering, Northwestern University, Evanston, IL 60208

DOI 10.1002/aic.14022
Published online February 7, 2013 in Wiley Online Library (wileyonlinelibrary.com)

An efficient decomposition method to solve the integrated problem of scheduling and dynamic optimization for sequential batch processes is proposed. The integrated problem is formulated as a mixed-integer dynamic optimization problem or a large-scale mixed-integer nonlinear programming (MINLP) problem by discretizing the dynamic models. To reduce the computational complexity, we first decompose all dynamic models from the integrated problem, which is then approximated by a scheduling problem based on the flexible recipe. The recipe candidates are expressed by Pareto frontiers, which are determined offline by using multiobjective dynamic optimization to minimize the processing cost and processing time. The operational recipe is then optimized simultaneously with the scheduling decisions online. Because the dynamic models are encapsulated by the Pareto frontiers, the online problem is a mixed-integer programming problem which is much more computationally efficient than the original MINLP problem, and allows the online implementation to deal with uncertainties. © 2013 American Institute of Chemical Engineers *AIChE J*, 59: 2379–2406, 2013

Keywords: integrated scheduling and dynamic optimization, decomposition, mixed-integer dynamic optimization, mixed-integer nonlinear programming, multiobjective dynamic optimization

Introduction

Confronting current challenges in process industries, an enterprise has to optimize its overall production activities including design, planning, scheduling, and dynamic optimization.[1–4] Integrating decision making at different levels becomes of utmost importance to identify economic potentials for increasing the profit margins which is being dwindled under increasingly fierce global competitions. A great variety of techniques have been developed for integration of planning and scheduling,[5–11] design and scheduling,[12–14] design and control,[15–19] and scheduling and dynamic optimization.[20–26] Comparing with other integration strategies, integration of scheduling and dynamic optimization is more challenging because there are constraints of nonlinear differential equations in the integrated problem and it needs to be resolved online under uncertainties.[20,27] The objective of this work is to develop a fast computational method to solve the complicated problem of online integrated scheduling and dynamic optimization for batch processes.

The integrated problem is generally formulated as a mixed-integer dynamic optimization (MIDO)[28,29] problem where the differential equations describing the process dynamics are incorporated into the scheduling model with discrete and continuous variables. The MIDO problem can be solved by the simultaneous method[30,31] where the MIDO problem is reformulated into a mixed-integer nonlinear programming (MINLP) problem by discretizing the differential equations into nonlinear algebraic equations. Though the formulation is straightforward, solving the formulated large-scale MINLP is challenging.[20] For a multiproduct, multistage batch process, there are a number of dynamic models describing each operational stage for a product. Moreover, the number of dynamic models increases with the number of the batches. A dynamic model needs to have different copies, one for each batch. As a result, an integrated problem is generally a large-scale MINLP even though the constituent scheduling problem might be simple.

Comparing with the conventional approach, the advantage of an integrated method is that the operational recipe can be optimized simultaneously with scheduling decisions. Conventionally, scheduling and dynamic optimization are solved in a sequential way. This is the sequential method which determines the recipe data for scheduling, for example, batch processing times, by optimizing operations of a single product. Then, the schedule is optimized based on the fixed recipe. However, in a batch-scheduling problem, a product competes with others for resources, for example, materials, equipment, and storages. Due to the interactions among products, the recipe, which is optimal for a single product, might be suboptimal for the entire scheduling problem of a multiproduct batch process. Integration of scheduling and dynamic optimization is essential to optimize the overall performance of the batch production by determining the optimal recipe simultaneously with the scheduling decisions.[32]

The integrated optimization problem is, however, much more challenging to solve than the sequential optimization approach. Even if the scheduling model is linear, the integrated problem becomes nonlinear after incorporating the

Correspondence concerning this article should be addressed to F. You at you@northwestern.edu.

dynamic models. Due to the complexity of the formulated MINLP problem, the optimal solution is difficult to obtain even for offline optimization.[33,34] To deal with uncertainties which are inevitable in the batch production, the integrated problem needs to be resolved online. The online optimization is much more challenging because the optimal solution should be found in a short-time period. The main challenges in an integrated method are summarized as

- Complicated MINLP problem resulting from the incorporation of dynamic models into the scheduling problem with discrete decisions.
- High requirement on computational efficiency for online optimization in a short period under uncertainties.

To address these challenges, we propose a novel framework to solve the integrated scheduling and dynamic optimization problem. It is applied to sequential batch processes where the batch integrity can be preserved. The proposed method consists of several stages shown in Figure 1. (1) An integrated problem is formulated by extending the time-slot scheduling model[35] and the order price function.[36] (2) A bilevel optimization approach is applied to analyze the model structure and identify the conflicting factors of the processing times and the processing costs which link the scheduling model and the dynamic models. (3) The Pareto frontiers of the conflicting factors are obtained by the multi-objective, multistage dynamic optimization for each product. The ε-constraint approach is applied to determine the Pareto frontiers. (4) The Pareto frontiers are used to encapsulate the dynamic models and represent the flexible recipe for scheduling. The integrated problem is decomposed into dynamic optimization problems, which are solved offline to generate a flexible recipe, and a scheduling problem based on the flexible recipe, which is solved online. There are two alternatives to build the online scheduling model: (i) a continuous-time scheduling model based on piecewise linear Pareto curves and (ii) a discrete-time scheduling model based on discrete Pareto points. Because all dynamic models are separated, the online scheduling problem becomes a mixed-integer linear programming (MILP) problem. (5) The scheduling decisions and the operational recipes are optimized simultaneously online to deal with uncertainties in the batch production.

The novelties of the proposed framework are summarized as

- New integrated model: extend both continuous-time scheduling model and discrete-time scheduling model to formulate a compact integrated problem which incorporates the minimum number of dynamic models.
- Efficient solution strategy: decompose all dynamic models from the integrated problem and approximate the integrated problem by an MILP scheduling model with flexible recipe, avoiding the solution of a complicated MINLP problem.
- Capable for online application: All dynamic optimization problems are solved offline and the scheduling model with flexible recipe is computationally efficient and suitable for online implementation under uncertainties.

The remaining of this article is organized as follows. The problem formulation is given in Problem Formulation. The decomposition strategy is presented in Solution Strategies. The section Case Study includes a detailed study on a flexible jobshop scheduling problem with dynamic models of reaction tasks. Conclusions are drawn at the end.

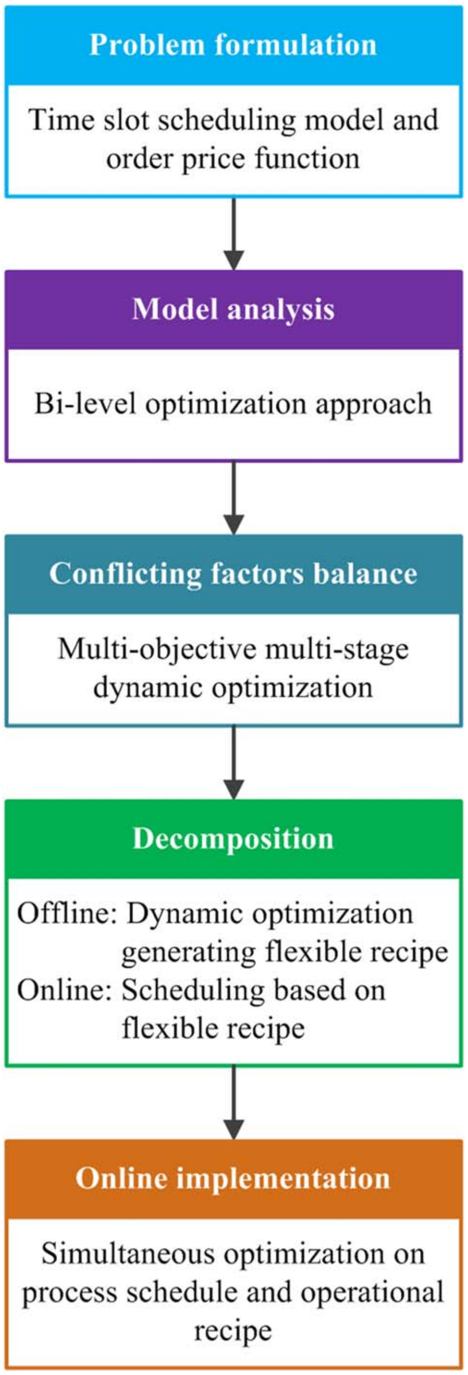

Problem formulation
Time slot scheduling model and order price function
Model analysis
Bi-level optimization approach
Conflicting factors balance
Multi-objective multi-stage dynamic optimization
Decomposition
Offline: Dynamic optimization generating flexible recipe
Online: Scheduling based on flexible recipe
Online implementation
Simultaneous optimization on process schedule and operational recipe

Figure 1. Integration framework of scheduling and dynamic optimization for sequential batch processes.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Problem Formulation

There are various scheduling models for batch processes with different specifications. We need to first select a scheduling model to formulate the integrated problem. The scheduling model determines the number of dynamic models incorporated in the integrated problem. This work focuses on the sequential process and extends the time-slot scheduling model[35] to formulate the integrated problem. The integrated problem based on the time-slot model contains fewer dynamic models than the integrated problems formulated

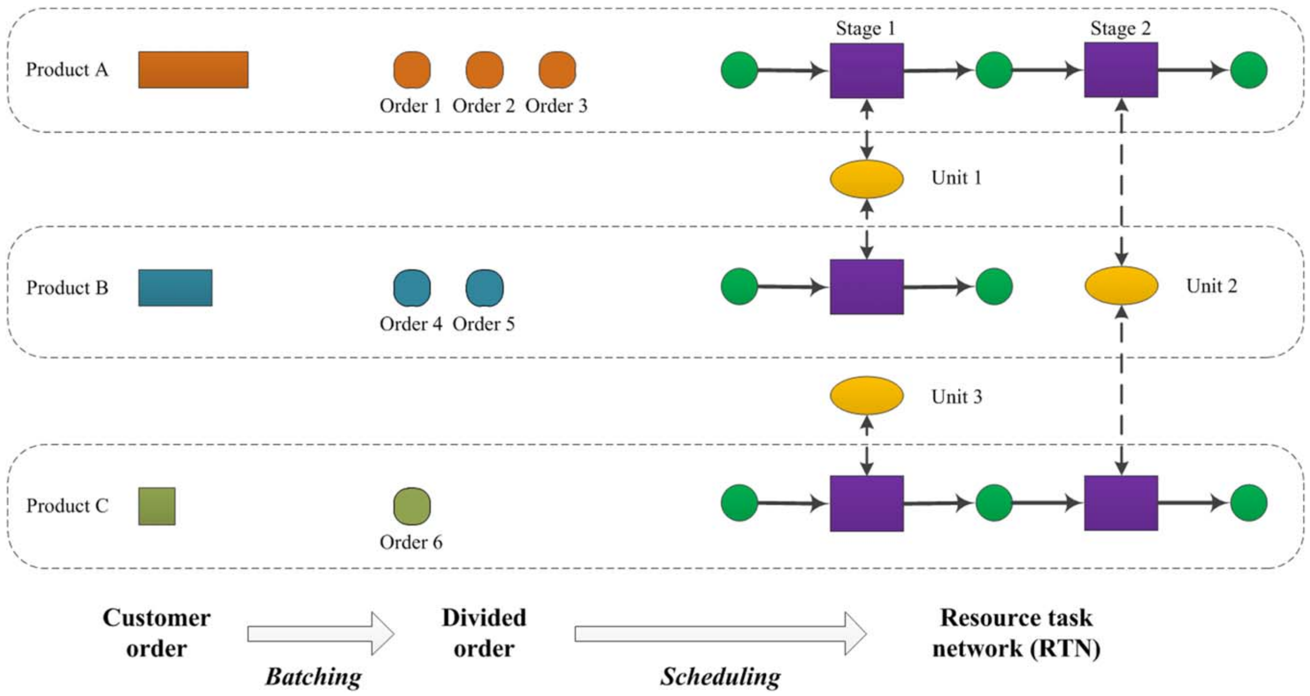


Figure 2. Diagram of scheduling a sequential multiproduct, multiorder, multistage batch process.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

based on the unit-specific continuous-time model[34] and the state-equipment network (SEN) model.[33]

Based on the time-slot model, the number of dynamic models included in the integrated problem is independent on the number of time points. This feature makes it possible to extend the discrete-time scheduling model to formulate the integrated problem. Both continuous-time scheduling model and the discrete-time scheduling model have advantages and disadvantages.[37–40] The performance is highly dependent on the problem specifications. Therefore, it is desirable to use both scheduling models and make a comparison if possible. We note that most existing methods for the integration of scheduling and dynamic optimization use the continuous-time scheduling model and no comparison with the discrete-time scheduling model is reported in the literature.

Scheduling problem of sequential batch processes

Batch processes are commonly cast into two categories: the network process and the sequential process.[37,41] The network process is general where operations of batch splitting, mixing, and resizing are allowed. The sequential process is special where batch splitting, mixing, and resizing are prohibited. These restrictions result in a feature of the sequential process that the batch integrity is preserved through production stages. Though methods for the network process are also applicable to the sequential process, they are much less efficient than the specific methods taking advantage of the special process structure.[42] Because the scheduling problem of a process with a general network structure can be challenging to solve online by itself, this work focuses on the sequential process to maintain the mathematical tractability on a problem with a reasonable size. Studies on the general network scheduling problem are future work.

An illustrative diagram of scheduling the sequential batch process is given in Figure 2. Products A, B, and C are manufactured. The customer order specifies the demand and the due date for each product. Because the batch integrity is preserved in the sequential process, the number of batches can be determined according to the customer demand. This is the batching procedure,[35] a common step in scheduling the sequential process.

As illustrated in Figure 2, the original order is divided into smaller orders. Each divided order is fulfilled by one batch of the associated tasks. When all divided order is satisfied, the original order is satisfied. Thus, we only need to track the fulfillment of each divided order and the original demand constraints can be replaced by the order fulfillment constraint. The divided order plays an important role in the following mathematical formulation. To help understand the order, we can make an analogy between the sequential batch process scheduling problem and the jobshop scheduling problem. The term "order" corresponds to the term "job" in the jobshop problem, whereas the term "stage" is the counterpart of the term "operation".[43]

The routing of an order execution can be represented by the resource task network (RTN).[44] There are three types of nodes: (1) the task nodes (represented by rectangles), which denote the operational stages; (2) the state nodes (represented by circles), which denote the raw materials, intermediate products, and the final products; (3) the equipment nodes (represented by ellipses), which denote the processing units. Though the routing of a product order can be separated from those for other products, the operational stages for different products can compete for the same unit. The shared resources of processing units impose interactions among different orders. Therefore, the scheduling problem should be solved by taking all orders simultaneously into account.

The goal of scheduling is to optimize an economic criterion by determining the starting time of an operational stage for an order and allocating a processing unit to an operational stage. These two types of decisions can be modeled at

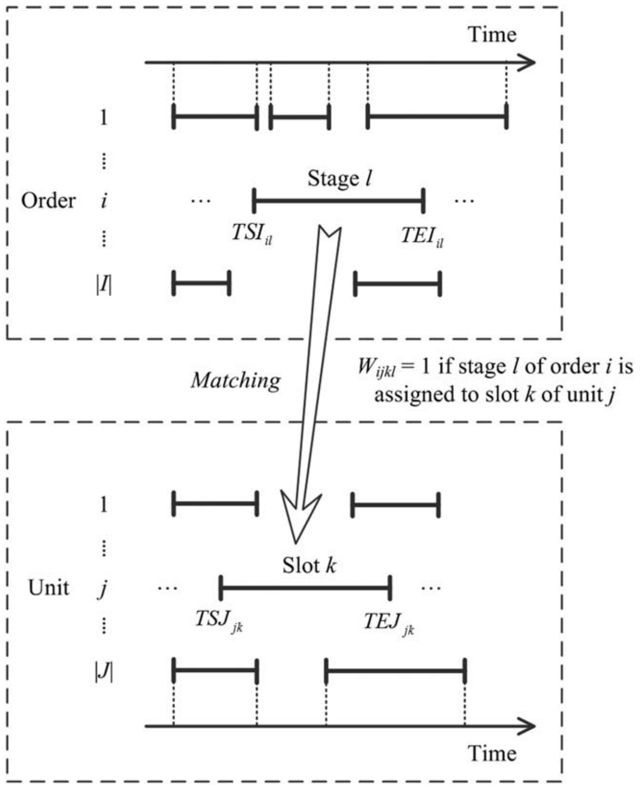


Figure 3. Time slot model.

The main scheduling decision is to assign an order stage to a unit slot.

the same time by using the time-slot formulation.[35] An illustration of the time-slot model is shown in Figure 3. An order indexed by i has multiple operational stages indexed by l. A unit indexed by j has a set of time slots indexed by k. The starting time of a stage is denoted by TSI_{il} while the ending time is TEI_{il}. To process different orders, the time interval of a unit is partitioned into time slots. Each unit processes an order only in a time slot. The starting time and the ending time of a slot are denoted by TSJ_{jk} and TEJ_{jk}, respectively. The allocation of units is determined by a matching operation between an operational stage and a time slot. A binary variable W_{ijkl} is introduced to model the matching decisions. If $W_{ijkl}=1$, then the operational stage indexed by (i, l) is matched to the time slot indexed by (j, k). Once they are matched, their starting times are equal, $TSI_{il}=TSJ_{jk}$.

Specifically, the time-slot model consists of the following constraints.[35]

a. Unit allocation

$$\sum_{j\in J_{il}}\sum_{k\in K_j} W_{ijkl}=1, \quad \forall i, \quad l\in L_i \tag{1}$$

$$\sum_{(i,l)\in IL_j} W_{ijkl}+S_{jk}=1, \quad \forall j, \quad k\in K_j \tag{2}$$

$$S_{jk-1}\le S_{jk}, \quad \forall j, \quad k\in K_j \quad \text{and} \quad k\ge 2 \tag{3}$$

$$S_{jk}\ge 0, \quad \forall j, \quad k\in K_j \tag{4}$$

where W_{ijkl} is the allocation variable and S_{jk} is a slack variable. Equation 1 imposes that an order is processed exactly once in each stage. The order indexed by i is the divided order as illustrated in Figure 2. When each divided order is fulfilled according to Eq. 1, the original order demand is satisfied. Set L_i contains all operational stages of order i, set J_{il} includes all units capable for processing stage l of order i, and set K_j contains the time slots in unit j. Units can have different numbers of time slots.

Equation 2 ensures the time slot of a unit is assigned to at most one stage of an order at a time. Set IL_j contains the pairs of order-stage indexed by (order, stage) that unit j can process. In the case of existing parallel units, empty time slots appear which are not matched to any stage of any order. An example of parallel units and empty time slots is displayed in Figure 4. When empty time slots appear, the slack variable S_{jk} in Eq. 2 is not zero. The slack variables are introduced to push all empty time slots to the end according to inequality (3) and they are all positive under inequality (4). As shown in Figure 4, when two time slots are available and only one is required, the first one is selected ($S_{j(k=1)}=0$) while the second one becomes empty ($S_{j(k=2)}=1$). The slack variables can help reduce the number of combinations for the time slot assignment and reduce the computational time.

b. Timing relation of operational stages

$$TEI_{il}=TSI_{il}+st_{il}+PT_{il}, \quad \forall i, \quad l\in L_i \tag{5}$$

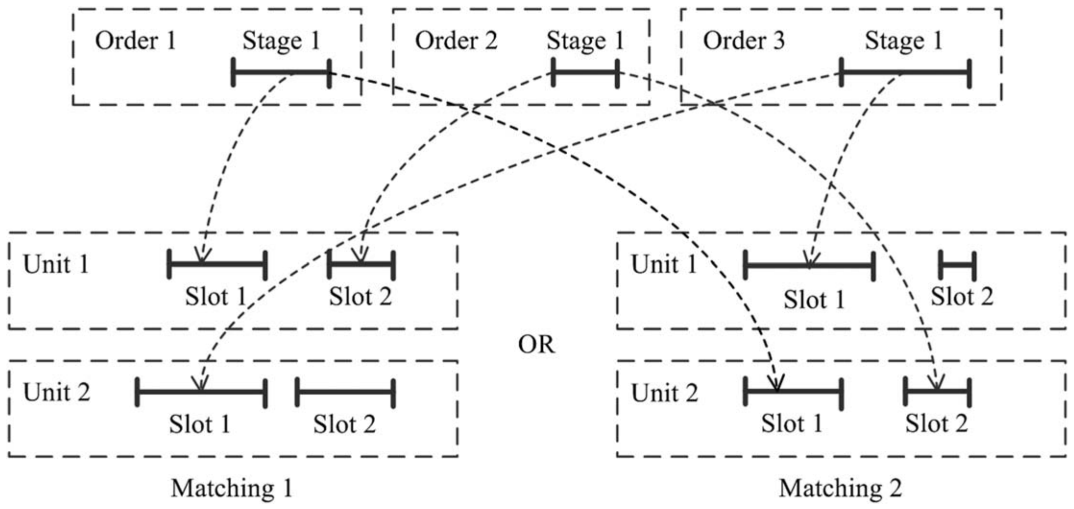


Figure 4. Parallel units and empty time slots.

$$TSI_{il} \geq TEI_{i(l-1)}, \quad \forall i, \quad l \in L_i, \quad l \geq 2 \tag{6}$$

Equation 5 defines the ending time TEI_{il} of a stage which is equal to the starting time TSI_{il} plus the setup time st_{il} and the processing time PT_{il}. The starting time of the current stage should be larger than or equal to the ending time of the previous stage. The relationship is modeled by inequality (6).

c. Timing relation of unit slots

$$TEJ_{jk} = TSJ_{jk} + \sum_{(i,l) \in IL_j} W_{ijkl}(st_{il} + PT_{il}), \quad \forall j, \quad k \in K_j \tag{7}$$

$$TSJ_{jk} \geq TEJ_{j(k-1)}, \quad \forall j, \quad k \in K_j, \quad k \geq 2 \tag{8}$$

Similar to the timing relations of an order stage, Eq. 7 states that the ending time TEJ_{jk} of a unit time slot is equal to the starting time TSJ_{jk} plus the processing time expressed via the allocation variables. The time slots of a unit cannot overlap so inequality (8) is placed on the starting time and the ending time of two adjacent slots.

d. Matching operational stage to time slot

$$TSI_{il} - TSJ_{jk} \geq -t_H(1 - W_{ijkl}), \quad \forall i, \quad j \in J_{il}, \quad k \in K_j, \quad l \in L_i \tag{9}$$

$$TSI_{il} - TSJ_{jk} \leq t_H(1 - W_{ijkl}), \quad \forall i, \quad j \in J_{il}, \quad k \in K_j, \quad l \in L_i \tag{10}$$

When an operational stage is processed in a time slot, the allocation variable $W_{ijkl}=1$. Then, the starting time of the stage is equal to the starting time of the slot according to inequalities (9) and (10), where t_H is the upper bound of the scheduling horizon. Once the starting times are matched, the finishing times are automatically matched according to Eqs. 5 and 7.

e. Sales according to order price functions

An important constraint on the batch scheduling problem is the order due date. An order needs to be fulfilled before its due date. Otherwise, a penalty is added to the tardy order and the corresponding selling price decreases with time. We introduce the order price function[36] into the time-slot scheduling model to evaluate an order according to its delivery date.

The order price function is shown in Figure 5. It is equal to the constant order price minus the tardiness penalty. The order price function is a piecewise linear function with three segments. The ending points are the first due date d_i^I, the second due date d_i^{II}, and the scheduling horizon t_H. The order can be delivered beyond the two due dates. However, the economic price of the order is less than the original price c_i^P if the delivery date exceeds the first due date d_i^I. The longer the order delivery is delayed, the larger penalty is placed on the tardiness and the less order price left. After the second due date d_i^{II}, the order would have zero price. The scheduling horizon t_H is a hard constraint on the order delivery date (ODT)

$$ODT_i = TEI_{i(l=|L_i|)}, \quad \forall i \tag{11}$$

$$ODT_i = DT_i^I + DT_i^{II} + DT_i^{III}, \quad \forall i \tag{12}$$

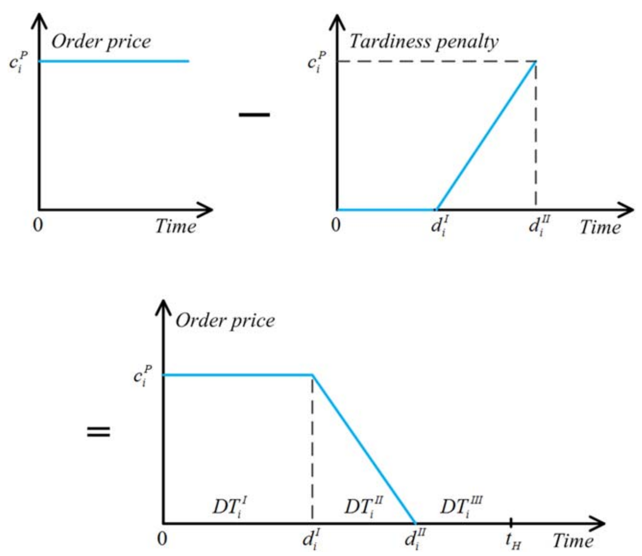


Figure 5. The order price function is equal to the constant order price minus the tardiness penalty.

The order price function is characterized by the order price c_i^P, the first due date value d_i^I, the second due date value d_i^{II}, and the scheduling horizon t_H. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

$$OW_i^I d_i^I \leq DT_i^I \leq d_i^I, \quad \forall i \tag{13}$$

$$OW_i^{II}(d_i^{II} - d_i^I) \leq DT_i^{II} \leq OW_i^I(d_i^{II} - d_i^I), \quad \forall i \tag{14}$$

$$DT_i^{III} \leq OW_i^{II}(t_H - d_i^{II}), \quad \forall i \tag{15}$$

$$\text{Sales} = \sum_i c_i^P \left(1 - \frac{DT_i^{II}}{d_i^{II} - d_i^I}\right) \tag{16}$$

The ODT_i of order i is defined by the ending time of the last stage according to Eq. 11. In the order price function, the delivery date is expressed as the sum of components DT_i^I, DT_i^{II}, and DT_i^{III} in Eq. 12. The piecewise order price function is expressed by Eqs. 13–15, where the binary variables OW_i^I and OW_i^{II} are introduced to express the location of the delivery date. Combinations of OW_i^I and OW_i^{II} denote the intervals in which the ODT_i stays, that is,

$$OW_i^I = 0, \quad OW_i^{II} = 0 \Rightarrow DT_i^{II} = 0, \quad DT_i^{III} = 0 \Rightarrow 0 \leq ODT_i \leq d_i^I$$

$$OW_i^I = 1, \quad OW_i^{II} = 0 \Rightarrow DT_i^I = d_i^I, \quad DT_i^{III} = 0 \Rightarrow d_i^I \leq ODT_i \leq d_i^{II}$$

$$OW_i^I = 1, \quad OW_i^{II} = 1 \Rightarrow DT_i^I = d_i^I, \quad DT_i^{II} = d_i^{II} - d_i^I \Rightarrow d_i^{II} \leq ODT_i \leq t_H$$

The order price is expressed in Eq. 16 and the sum defines the sales of all orders.

f. Production cost

$$\text{Cost} = \sum_i (C_i^{Var} + c_i^{Fix}) = \sum_i C_i^{Var} + c^{Fix} \tag{17}$$

The total production cost is the sum of the costs for every order i. The cost of order i consists of two terms: the variable cost C_i^{Var} and the fixed cost c_i^{Fix}. The sum of the fixed costs over all orders is denoted by c^{Fix}. The variable cost is expressed by the sum of the costs over all stages as

$$C_i^{Var} = \sum_{l \in L_i} PC_{il}, \quad \forall i \tag{18}$$

where PC_{il}, is the processing cost of stage l for order i. The processing cost is determined by the dynamic model in the operational stage and the detailed expression will be discussed in the following sections.

g. Objective function of profit

$$\text{Profit} = \text{Sales–Cost} \tag{19}$$

The objective of the scheduling problem is to maximize the production profit which is equal to the sales minus the cost. The sales are solely determined by the order price function in Eq. 16 and they are not dependent on the product quality. The products are demanded with a customer-specified quality level. The delivered products should satisfy the quality requirement. However, the customers will not pay more for a high-quality product beyond what they requested.

Multistage dynamic optimization

A feature in integration of scheduling and dynamic optimization is that the detailed dynamic models are taken into account in the scheduling problem. Unlike the conventional scheduling method where the recipe data such as processing times are fixed parameters, an integrated method uses a flexible recipe. The recipe data become variables which are determined by dynamic models in operational stages of an order. We can change the flexible recipe by manipulating process inputs.

For batch production, a product is commonly manufactured by a series of cascaded operational stages rather than a single stage only (e.g., see Figure 2). The initial condition of a dynamic model in a stage is dependent on the final value of the dynamic model in the previous stage. Therefore, the cascaded dynamic models for an order need to be optimized simultaneously so that the initial condition or the final value linking two models can be regarded as decision variables. This is a feature different from conventional dynamic optimization which studies the model in a single unit.

A dynamic model in an operational stage can be described by a set of differential algebraic equations

$$0 \le t_{il} \le \tau_{il}, \quad \forall i, \quad l \in L_l \tag{20}$$

$$\frac{d}{dt_{il}} x_{il}(t_{il}) = f_{il}(t_{il}, x_{il}(t_{il}), u_{il}(t_{il})), \quad \forall i, \quad l \in L_l \tag{21}$$

$$h_{il}(t_{il}, x_{il}(t_{il}), u_{il}(t_{il})) \le 0, \quad \forall i, \quad l \in L_l \tag{22}$$

$$y_{il}(t_{il}) = g_{il}(t_{il}, x_{il}(t_{il}), u_{il}(t_{il})), \quad \forall i, \quad l \in L_l \tag{23}$$

$$\varphi_{il} = \phi_{il}(y_{il}(\tau_{il})), \quad \forall i, \quad l \in L_l \tag{24}$$

$$x_{il}(0) = \begin{cases} x_{i(l-1)}\left(\tau_{i(l-1)}\right), & l \ge 2 \\ x_i^0, & l=1 \end{cases}, \quad \forall i, \quad l \in L_l \tag{25}$$

$$x_i^f = x_{il}(\tau_{il}), \quad l = |L_i|, \quad \forall i \tag{26}$$

$$q_i\left(x_i^f\right) \le 0, \quad \forall i \tag{27}$$

The dynamic model is identified by the order index i and the stage index l. In this work, we consider identical parallel units so that the dynamic model is the same in those parallel units. If the units for a stage are not identical, then we need consider different combinations. For example, we assume unit A for the first stage and unit B or C for the second stage. If unit B is not identical to unit C, then the multistage optimization problem on the two stages needs to be solved for each combination of units, A, B or A, C.

We should note that the purpose of the dynamic optimization is to generate the candidate recipe data for the scheduling problem. Though we need to solve more dynamic optimization problems for different combinations of the nonidentical units, all dynamic optimization problems are solved offline. The presence of the nonidentical units will not affect the efficiency of the proposed method, which will be presented in the following section.

Variables in the dynamic model are labeled in the same way, including the time variable. The time t_{il} is restricted by inequalities (20). The starting time is 0 and the ending time is τ_{il}. Equation 21 consists of differential equations of the state variables x_{il} and input variables u_{il} . Path constraints (22) on the state variables and the input variables contain a set of inequalities, expressing the safety constraints, the utility constraints, or other general constraints on the process.

The output equations are expressed by Eq. 23. In an integrated problem, the dynamic process is optimized to improve some economic criterion. The outputs y_{il} are related with the criterion value φ_{il} in Eq. 24. The criterion function is evaluated at the final time point of the output. This expression is general to model the criterion related with the entire time function. For example, if we need to calculate the accumulated input value in the entire time interval, that is

$$\int_0^{\tau_{il}} u_{il}(t_{il})dt_{il}$$

we can append the following differential equation

$$\frac{d}{dt_{il}} z_{il}(t_{il}) = u_{il}(t_{il})$$

to the dynamic model (21). Next, we define the output as

$$y_{il}(t_{il}) = z_{il}(t_{il})$$

and the criterion function as

$$\varphi_{il} = y_{il}(\tau_{il})$$

Then, we have the value we need

$$\varphi_{il} = \int_0^{\tau_{il}} u_{il}(t_{il})dt_{il}$$

Equation 25 defines the initial condition of each dynamic model, which is equal to the final value of the dynamic model in the previous stage. If the stage is the first one, the initial value is equal to the parameter x_i^0 dependent on product i, for example, the species concentrations in the raw materials. The value of the final product is modeled by Eq. 26, which is equal to the final value in the final stage. The final value x_i^f denotes the product quality, for example, the species concentrations in the final product. We can place quality constraints (27) on the final value.

The linking variables between the scheduling model and a dynamic model are the processing time τ_{il} and the processing cost φ_{il}. The two variables are coupled by the dynamic model and manipulated by the process inputs. Both variables also affect the objective function of the scheduling problem. The processing time affects the ODT and in turn the order price. The processing cost is a component of the total cost. Due to the linking variables, the scheduling problem and the dynamic optimization problem should be solved in an integrated way. It should be noted that the batch size is determined in the batching procedure before solving the scheduling problem. Thus, the batch size is not a linking variable, but a fixed parameter for both scheduling problem and dynamic optimization problem.

Constraints of the differential Eqs. 21 cannot be handled directly by a standard nonlinear programming (NLP) solver. Thus, the differential equations are often discretized into algebraic equations. First, the continuous time interval is expressed by a set of discrete time points, $\left\{t_{il}^{(n)}\right\}$, $n=1, 2, \ldots, n_f$. The step between two adjacent points is defined by $\sigma_{il}=t_{il}^{(n+1)}-t_{il}^{(n)}$. Then, the continuous-time functions are all sampled at the discrete time points, that is, $x_{il}^{(n)}=x_{il}\left(t_{il}^{(n)}\right)$, $u_{il}^{(n)}=u_{il}\left(t_{il}^{(n)}\right)$, and $y_{il}^{(n)}=y_{il}\left(t_{il}^{(n)}\right)$. Finally, the differential equations which express relations among the continuous-time functions are transferred into algebraic equations which place constraints on the discrete points, as

$$x_{il}^{(n)}=x_{il}^{(n-1)}+\sum_{p=1}^{N_p} b_p k_{il}^{(p)} \tag{28}$$

$$k_{il}^{(p)}=\sigma_{il} f_{il}\left(t_{il}^{(n)}+c_p\sigma_{il}, x_{il}^{(n)}+\sum_{q=1}^{N_p} a_{pq} k_{il}^{(q)}, u_{il}^{(n)}\right) \tag{29}$$

where a_{pq}, b_p, and c_p are parameters, defined by a specific discretization method. Equations 28 and 29 comprise the general expression of Runge–Kuta methods, including both explicit methods ($a_{pq}=0$, for any $q\geq p$) and implicit methods ($a_{pq}\neq 0$, for a $q\geq p$). Runge–Kuta methods are multistage methods and the integer n_p denotes the number of stages.

Equations 28 and 29 can also represent the formulation of collocation methods,[45] where $x_{il}^{(n)}$ denotes a finite element point and $k_{il}^{(n)}$ denotes a collocation point. Collocation methods use a piecewise polynomial to approximate the solution of the differential equations. In fact, a collocation method with the polynomial of the n_p th degree is equivalent to an implicit Runge–Kutta method with n_p stages.[46]

After the time variable is discretized, other equations describing the dynamic model become constraints at the discrete time points, as shown in the equations and inequalities below

$$h_{il}\left(t_{il}^{(n)}, x_{il}^{(n)}, u_{il}^{(n)}\right)\leq 0 \tag{30}$$

$$y_{il}^{(n)}=g_{il}\left(t_{il}^{(n)}, x_{il}^{(n)}, u_{il}^{(n)}\right) \tag{31}$$

$$\varphi_{il}=\phi_{il}\left(y_{il}^{(n=n_f)}\right) \tag{32}$$

$$x_{il}^{(n=1)}=\begin{cases} x_{i(l-1)}^{(n=n_f)}, & l\geq 2 \\ x_i^0, & l=1 \end{cases} \tag{33}$$

$$x_i^f=x_{il}^{(n=n_f)}, \quad l=|L_i| \tag{34}$$

Note that we drop the il indices for the discretization parameters, n_f, a_{pq}, b_p, c_p, and n_p, for simplicity. The dynamic models can be discretized by using different parameter values.

Formulation of integrated problem

Scheduling problem requires the process recipe as the input data. The processing times and the processing costs are determined by the dynamic models in the order operational stages. For a traditional scheduling method, the recipe consists of fixed parameters which are determined by solving dynamic optimization problems in advance. A main drawback of traditional methods is that it cannot account for the objective function of the scheduling problem. The dynamic optimization problem can only optimize some local objective function, for example, minimizing the processing time. Therefore, the recipe determined by the dynamic optimization might not be optimal for the scheduling problem. In the integrated approach, this problem is solved by determining the optimal recipe simultaneously with the optimal scheduling decisions.

The integrated problem we analyze is stated as follows

Assumptions:
- Sequential batch process where there are no batch splitting, mixing, and resizing

Given:
Scheduling parameters
- Production configuration by RTN, including the routing pathway for executing an order and the information of capable units for an operational stage
- Upper bound of the scheduling horizon
- Divided orders with due dates
- Price and fixed cost for each order
- Unit cost of raw materials and utilities

Process dynamics
- Dynamic models of operational stages
- Cost function related to a dynamic model
- Initial condition in the first stage and final value in the last stage associated with each order
- Utility constraints, safety constraints, and quality constraints

Determine:
Scheduling decisions
- Starting time of each operational stage of an order
- Allocation of a processing unit to each stage of an order
- Delivery date and economic value of an order

Recipe
- Processing time and processing cost of each operational stage

Objective:
- Maximize the production profit

The integrated problem is formulated by combining the scheduling model and the dynamic models, which is

$$\max \quad \text{Profit (Eq. 19)} \tag{35}$$

s.t.
Scheduling model (Eqs. 1–18)
Dynamic models (Eqs. 20–27)

$$PT_{il}=\tau_{il}, \quad \forall i, \quad l\in L_i \tag{36}$$

$$PC_{il}=\varphi_{il}, \quad \forall i, \quad l\in L_i \tag{37}$$

This is an MIDO problem with constraints including differential equations. The scheduling model is linked to the

Table 1. Number of Dynamic Models in the Integrated Problem for Different Continuous-Time Scheduling Models

Scheduling Model	Time-Slot Model in This Work	Unit-Specific Continuous-Time Model[34]	State-Equipment Network Model[33]
Number of dynamic models	#Tasks	#Task × #Time points	#Unit states × #Units × #Time points

dynamic models by the equation of the processing times (36) and the equation of the processing costs (37). The MIDO problem can be discretized into an MINLP problem by replacing Eqs. 20–26 with Eqs. 28–34. Although it is straightforward to reformulate the MIDO problem into an MINLP, solving the reformulated MINLP could be computationally challenging.[47,48]

The number of dynamic models is a factor indicating the complexity of the integrated problem, which is determined by the scheduling model. Table 1 lists the number of dynamic models for different continuous-time scheduling models. Mishra et al.[34] adopts the unit-specific continuous-time model.[49] In the formulation, a dynamic model is identified by the indices of tasks and time points. Thus, the number of dynamic models in the integrated problem is equal to the number of tasks times the number of time points. Nie et al.[33] uses the SEN and the number of dynamic models is the product of the number of unit states, the number of units, and the number of time points. For the time-slot model extended in this work, the number of dynamic models in the integrated problem is equal to the number of tasks which is denoted by an order stage in the sequential process. Given that each task has a unique dynamic model different from those for other tasks, the number of dynamic models included in an integrated problem cannot be fewer than the number of tasks. This means the integrated problem based on the time-slot scheduling model has the minimum number of dynamic models.

The number of dynamic models based on the time-slot model is not only smaller than other formulations but also independent on the time points. The number of time points is a critical parameter in a continuous-time scheduling model.[37] Unfortunately, it is often selected by trial and error due to a lack of a systematical selection procedure. So it is desirable to decouple the number of dynamic models from the number of time points, reducing the difficulty in setting up the integrated problem. The independence also makes it possible to extend the discrete-time scheduling model for the integrated problem even though the discrete-time scheduling model generally includes much more time points than the continuous-time scheduling model. Given that continuous-time scheduling model and discrete-time scheduling model both have pros and cons, both formulations can be used for the scheduling model of the integrated problem. Because the discrete time model has a much simpler structure than the continuous-time model, the computational requirement of the discrete-time model might not increase with the incorporation of additional features.[39]

Of course, the time-slot model can only be used to solve the sequential process scheduling problem. Consequently, the proposed method confines the integrated problem to the sequential process. The unit-specific continuous-time model and the SEN model can be applied to a more general network process which allow batch splitting and mixing. However, these models result in a much more complex integrated problem, consisting of a great number of dynamic models. Thus, application of the integrated method to a network process concentrates on small-scale problems.[33,34] We should note that the number of dynamic models cannot be automatically reduced when these methods are applied to the sequential process. Both existing methods use the simultaneous approach to solve the integrated problem. Even for the sequential process, the simultaneous approach could be very inefficient, preventing the online implementation. The proposed method, however, includes a decomposition approach, which can considerably reduce the computational time and ensure the online application. The decomposition method is presented in the following section.

Solution Strategies of Integrated Problem

The integrated problem is a complicated MIDO problem or a large-scale MINLP problem after the discretization procedure. Solving such a complicated problem is challenging. Furthermore, we need to take into account the uncertainties in a batch process. Batch production is subject to various uncertainties, for example, equipment breakdown, batch failure, processing time variation, and sudden change in the order. Under uncertainties, the production can soon deviates from the predetermined schedule and the initially optimal schedule may become suboptimal or even infeasible. Therefore, we often need to reschedule the process online to deal with uncertainties. Online implementation makes the solution to the integrated problem much more challenging. The integrated problem should be solved repeatedly according to the updated information and the solution should be obtained in a small-time interval so that uncertainties can be handled immediately.

In this section, we propose an efficient decomposition method to solve the integrated problem. First, we decompose the dynamic models from the integrated problem through a bilevel programming approach. To approximate the optimal-value function in the bilevel programming problem, we develop a heuristic method based on the multiobjective optimization on the conflicting factors of processing times and processing costs. The resulting Pareto curve provides a flexible recipe. The integrated problem is then transformed into a scheduling problem with the flexible recipe. Two types of models are derived based on different scheduling models and different expressions of the Pareto curve: (1) the continuous-time scheduling model based on piecewise linear Pareto curve and (2) the discrete-time scheduling model based on discrete Pareto points.

Decomposition based on bilevel optimization

The integrated problem is computationally challenging. The difficulty arises from the combination of the scheduling problem with the dynamic optimization problems. However, if we consider them separately, the individual problems are much easier to solve. This motivates us to develop a decomposition method which can separate the dynamic optimization problems from the scheduling problem in the optimization procedure. However, the dynamic information is taken into account in the flexible recipe which is optimized simultaneously with scheduling decisions. We apply a bilevel programming method to achieve this goal.

For the purpose of the decomposition, we first cluster the decision variables into four categories: (1) the variables only included in the scheduling problem, denoted by the collection V^{Sch}, (2) the variables only contained in the dynamic model of order i, denoted by the collection V_i^{Dyn}, (3) the collection of processing times $\{PT_{il}\}$, and the collection of processing costs $\{PC_{il}\}$. Using the classification, we can express the objective function of the integrated problem as

$$\text{Profit}^* = \max_{\substack{\{PT_{il}\},\forall i, l\in L_i \\ \{PC_{il}\},\forall i, l\in L_i \\ V^{Sch} \\ \left\{V_i^{Dyn}\right\},\forall i}} \text{Profit} \tag{38}$$

Then, we transform the optimization problem into a bilevel programming problem as

$$\text{Profit}^* = \max_{\substack{\{PT_{il}\},\forall i, l\in L_i \\ V^{Sch}}} \left\{ \max_{\substack{\{PC_{il}\},\forall i, l\in L_i \\ \left\{V_i^{Dyn}\right\},\forall i}} \{\text{Sales-Cost}\} \right\} \tag{39}$$

The processing times and the variables in the scheduling model are decision variables of the outer problem, whereas the processing costs and the variables in the dynamic models are decision variables in the inner problem. The profit is equal to the difference between the sales and the total cost. In the bilevel programming, the decision variables in the outer problem can be regarded as parameters in the inner problem. When we know the value of the processing times $\{PT_{il}\}$ and the scheduling variables in V^{Sch}, the term of the sales in the inner problem is independent on the processing costs or the variables in a dynamic model. We can minimize the cost term inside the inner problem

$$\text{Profit}^* = \max_{\substack{\{PT_{il}\},\forall i, l\in L_i \\ V^{Sch}}} \left\{ \text{Sales} - c^{Fix} - \min_{\substack{\{C_{il}\},\forall i, l\in L_i \\ \left\{V_i^{Dyn}\right\},\forall i}} \left\{ \sum_i \sum_{l\in L_i} PC_{il} \right\} \right\} \tag{40}$$

where the cost term is expressed as the sum of the processing costs plus the fixed cost according to Eq. 17. In the inner problem, the dynamic model of each order is independent with others. This is a very useful feature of the sequential process where there is no material splitting or mixing. According to the independence of dynamic models for different orders, we can further simplify the inner problem as

$$\text{Profit}^* = \max_{\substack{\{PT_{il}\},\forall i, l\in L_i \\ V^{Sch}}} \left\{ \text{Sales} - c^{Fix} - \sum_i \min_{\substack{\{C_{ij}\}, l\in L_i \\ V_i^{Dyn}}} \left\{ \sum_{l\in L_i} PC_{il} \right\} \right\} \tag{41}$$

where the dynamic optimization problem of each order can be solved independently because there is no material splitting or mixing for different orders in the sequential batch process.

Finally, we have the equivalent bilevel programming problem as

$$\max_{\substack{\{PT_{il}\},\forall i, l\in L_i \\ V^{Sch}}} \left\{ \text{Sales} - c^{Fix} - \sum_i P_i^C \right\} \tag{42}$$

s.t.

Scheduling model (Eqs. 1–18)

$$P_i^C = \eta_i\left(PT_{i1}, PT_{i2}, \cdots, PT_{i|L_i|}\right), \quad \forall i \tag{43}$$

Inner optimization for each order i

$$\eta_i\left(PT_{i1}, PT_{i2}, \cdots, PT_{i|L_i|}\right) = \min_{V_i^{Dyn}} \sum_{l\in L_i} PC_{il} \tag{44}$$

s.t.

Dynamic models (Eqs. 20–27)

$$PT_{il} = \tau_{il}, \quad \forall l \in L_i \tag{45}$$

$$PC_{il} = \varphi_{il}, \quad \forall l \in L_i \tag{46}$$

The bilevel programming approach reveals the underlying structure of the integrated problem, which is visualized in Figure 6. The inner problem includes the multistage dynamic optimization problem for each order. The objective is to minimize the total processing cost over all stages in Eq. 44. In the optimization, the processing times are regarded as parameters. Therefore, the optimal objective value is a function of the processing times, and it is denoted as the optimal-value function. The optimal-value function encapsulates the detailed dynamic models and it is a constraint in Eq. 43 for the outer problem. The outer problem is actually a scheduling problem based on a flexible recipe expressed by the optimal-value functions. If we have the expression of all optimal-value functions, we can easily solve the integrated problem by decomposing all dynamic models from the scheduling problem. We will discuss it in the following section.

The bilevel programming approach not only decomposes the complicated integrated problem, but also provides insights into the integration structure. A critical question is: can the integrated method outperform the sequential method? The sequential method solves the dynamic optimization problems separately before solving the scheduling problem, because the objective function of the scheduling problem is not available in the dynamic optimization. The sequential method uses some local information. It can minimize the

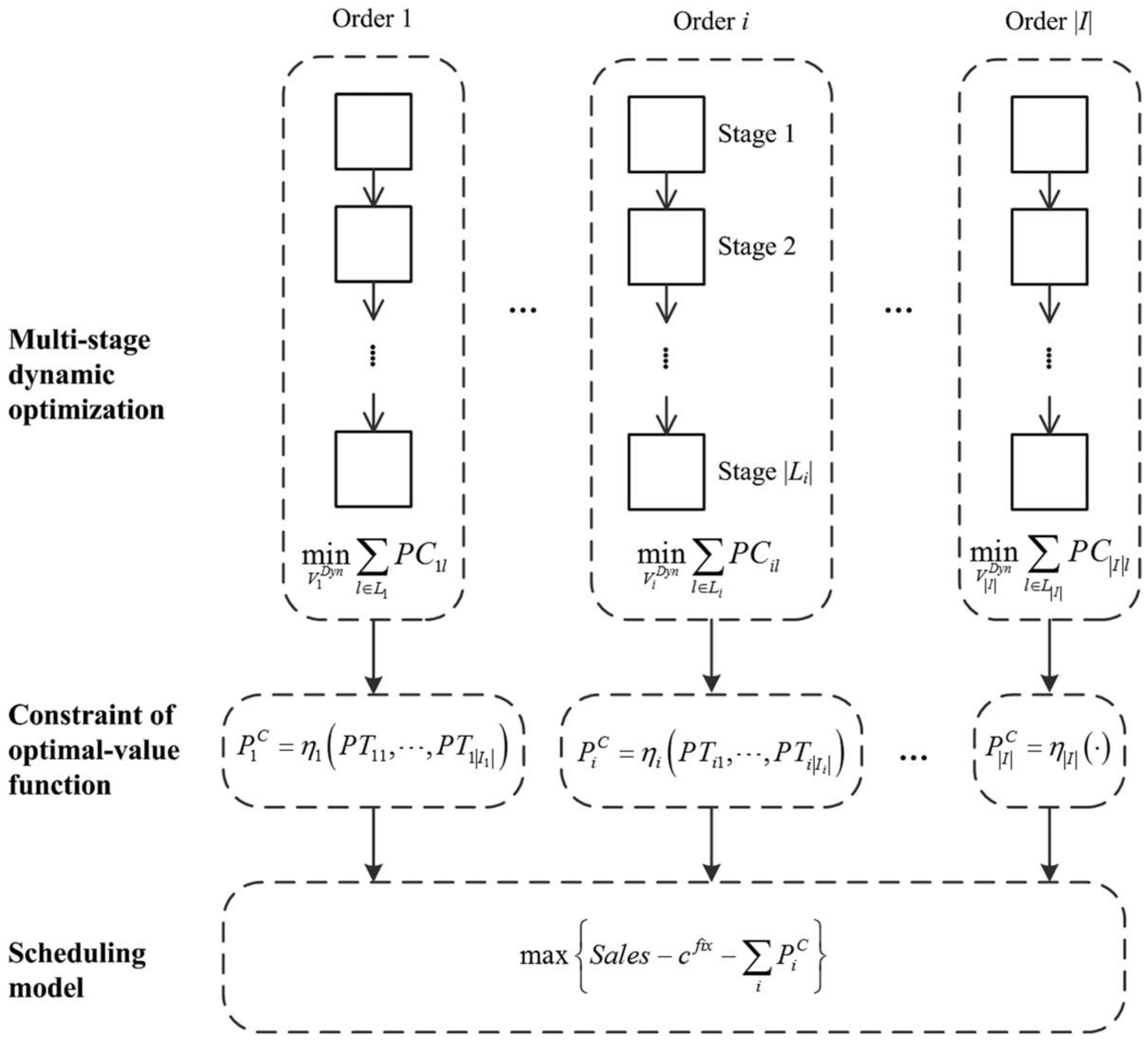

Order 1
Order i
Order $|I|$
Stage 1
Stage 2
Stage $|L_i|$
Multi-stage dynamic optimization
$\min_{V_1^{Dyn}} \sum_{l\in L_1} PC_{1l}$
$\min_{V_i^{Dyn}} \sum_{l\in L_i} PC_{il}$
$\min_{V_{|I|}^{Dyn}} \sum_{l\in L_{|I|}} PC_{|I|l}$
Constraint of optimal-value function
$P_1^C = \eta_1\left(PT_{11},\cdots,PT_{1|L_1|}\right)$
$P_i^C = \eta_i\left(PT_{i1},\cdots,PT_{i|L_i|}\right)$
$P_{|I|}^C = \eta_{|I|}(\cdot)$
Scheduling model
$\max\left\{Sales - c^{fix} - \sum_i P_i^C\right\}$

Figure 6. Structure of the integrated problem revealed by the bilevel programming.

processing time considering that a short processing time can reduce the ODT and increase the sales. A small value of the processing time is a beneficial factor for the profit. After the dynamic optimization, a fixed recipe is obtained. The scheduling problem is then solved based on the fixed recipe. Though the sequential method which only focuses on a single beneficial factor cannot guarantee a good overall performance, it provides a standard to evaluate the integrated method. From the structure of the integrated problem revealed by the bilevel programming, we can answer the aforementioned question. The positive answer relies on the condition that the processing times and the processing costs should be conflicting factors. It should be noted that the integrated method is not necessarily superior to the sequential method. Exploring the problem structure and identifying the conflicting factors are critical to determine if we really need to use such a complicated integrated method instead of a simple sequential one. Unfortunately, this issue is often neglected in literature.

To investigate why the processing time and the processing cost should be conflicting factors when an integrated method can achieve a better performance than the sequential one, we analyze the effects of the two types of variables in the scheduling problem. The processing times affect the sales by changing the order price function. When the processing time increases, it delays the ODT. Consequently, the order price (Figure 5) can only keep constant or decrease. The objective function of the profit monotonically (not strictly) decreases as the processing times increase. The processing costs affect the objective function by the cost term. As the total cost grows with the processing costs, the profit monotonically decreases as the processing costs increase.

In the scheduling problem, the processing times and the processing costs have the same effect on the tendency of the objective function. The two kinds of variables are, however, not dependent. They are coupled by the dynamic models. The coupling generates the optimal-value functions (43) which impose the constraints on the processing times and the processing costs. If the optimal-value function is a monotonically increasing function, then a small processing time implies a small processing cost. The optimal solution of the scheduling problem occurs at the point where the processing time is minimized. In this case, the integrated method might not outperform the sequential method. However, if the optimal value function is monotonically decreasing, the processing times and the processing costs become conflicting factors. When we shorten the processing time, the sales can increase. Meanwhile, the processing costs have to grow, resulting in a larger total cost. Balancing the two conflicting factors is not trivial and a good tradeoff can only be made when we solve the integrated problem.

Generation of promising recipe data from multiobjective optimization

In the integrated problem, the recipe data of processing times and processing costs are variables. Changing their values leads to a flexible recipe. The integrated problem can be regarded as an extended scheduling problem based on the flexible recipe. However, the processing times and the processing costs are coupled by the dynamic models. Their relationship is determined by constraints (43) which are the optimal-value functions (44) from the dynamic optimization problem. Because the optimal-value functions do not have closed-form expressions, we have to include all dynamic models into the integrated problem if we try to describe the entire set of recipe data according to constraints (43). This straightforward approach results in a very complicated integrated problem, challenging to solve.

To simplify the integrated problem, we develop a heuristic method which explores only a fraction of the recipe data set containing promising candidate recipes rather than the entire set. As discussed in the previous section, a short processing time or a small processing cost benefits the objective function. When they are conflicting factors, a short processing time implies a large processing cost and we cannot reduce both values simultaneously. Therefore, we need to consider different combinations of the processing times and the processing costs. For the conflicting factors, we can solve a multiobjective optimization problem and generate a set of promising recipe data from the returned Pareto frontier. The multiobjective optimization is solved for each order with the ε-constraint method

$$PT_{i1}(\varepsilon_i), \cdots, PT_{i|L_i|}(\varepsilon_i) = \arg \min_{PT_{il}, l\in L_i} \sum_{l\in L_i} PC_{il}, \quad \forall i \tag{47}$$

s.t.

Dynamic models (Eqs. 20–27)

Processing time and processing cost (Eqs. 45 and 46)

$$\sum_{l\in L_i} PT_{il} \le \varepsilon_i \tag{48}$$

The optimal solution of the processing times is a function of the parameter ε_i, which is the upper bound on the sum of the processing times. The recipe data for the processing costs are obtained as the optimal objective function value

$$P_i^C(\varepsilon_i) = \min_{PT_{il}, l\in L_i} \sum_{l\in L_i} PC_{il}, \quad \forall i \tag{49}$$

In the multiobjective optimization, we consider the tradeoff between the total processing time and the total processing cost. Therefore, we place the total processing time in the ε-constraint (48) and set the total processing cost as the objective function (47). The multistage dynamic optimization problem for an order is solved at each value of ε_i. The generated recipe data, processing times, and processing costs are one-dimensional (1-D) functions of the ε parameter. The single-variable function is straightforward to approximate.

Using the multiobjective optimization, we generate candidate recipe data from the Pareto frontier, which is parameterized by a single variable ε_i. Each processing time or processing cost is a function of ε_i. Substituting the parameterized recipe data into the integrated problem in Eqs. 42–46, we have

$$\max_{\substack{\{\varepsilon_i\}, \forall i \\ V^{Sch}}} \left\{ \text{Sales} - c^{Fix} - \sum_i P_i^C \right\} \tag{50}$$

s.t.

Scheduling model (Eqs. 1–18)

Recipe data for each order i

$$PT_{il} = \pi_{il}(\varepsilon_i), \quad \forall l \in L_i \tag{51}$$

$$P_i^C = \mu_i(\varepsilon_i) \tag{52}$$

where the functions π_{il} (ε_i) and μ_i (ε_i) represent the mapping from the Pareto frontier according to Eqs. 47–49. Though we do not have the closed-form analytical expression for the functions, we can simply approximate it using a piecewise linear function or a set of discrete points because they are 1-D functions. Based on the flexible recipe, all dynamic models are decomposed from the integrated problem.

The implementation of the integrated method is exhibited in Figure 7. In the offline implementation, the recipe functions $PT_{il} = \pi_{il}$ (ε_i) and $P_i^C = \mu_i(\varepsilon_i)$ are generated. We only need to investigate these functions for each product because orders for a product have the same production stages and the same optimal-value functions from the dynamic optimization. Even though the recipe function is the same, orders can have different operational conditions corresponding to different points selected from the same recipe function. We introduce the set I_P to represent the products. It is a subset of the order set $I_P \subseteq I$, in which only one order is preserved for each product. The multiobjective dynamic optimization problem is only solved on an order with $i \in I_P$. The retuned recipe function is applied to other orders of the same product.

Because the recipe functions are 1-D, the approximation is straightforward. The information we need for the approximation is a set of function values. We generate a set of discrete values $\left\{\varepsilon_i^{(r)}\right\}$, $\quad r = 1, 2, \cdots, n_i^\varepsilon$ for each product $i \in I_P$ where the number of the discrete values is denoted by n_i^ε. These values are incremental with the step size denoted by d_i^ε. The lower bound of the discrete points is determined by the dynamic optimization with the objective of minimizing the total processing time. Then starting from the lower bound, the set of discrete points is generated with the incremental step size. At each point of $\varepsilon_i^{(r)}$, the dynamic optimization problem in Eqs. 50–52 is solved. Then, the optimal processing times $PT_{il}^{(r)}$ and the optimal total processing cost $P_i^{C(r)}$ are recorded. From the set of $\left\{\varepsilon_i^{(r)}, PT_{il}^{(r)}\right\}$ and $\left\{\varepsilon_i^{(r)}, P_i^{C(r)}\right\}$, we can approximate the functions $PT_{il} = \pi_{il}$ (ε_i) and $P_i^C = \mu_i(\varepsilon_i)$, respectively. The details of the approximation will be discussed in the subsequent sections where different approximation approaches are combined with the continuous-time or discrete-time scheduling model. The approximated functions are denoted by $\tilde{\pi}_{il}(\varepsilon_i)$ and $\tilde{\mu}_i(\varepsilon_i)$.

In the online implementation in Figure 7, the optimal schedule is determined simultaneously with the optimal recipe. When uncertainties occur in the batch process, we measure the current production condition and then solve the extended scheduling with the flexible recipe problem according to the updated information. The reoptimized schedule and recipe are applied to the process to deal with uncertainties. A promising feature of the integrated method is that all dynamic optimization problems are solved online to determine flexible

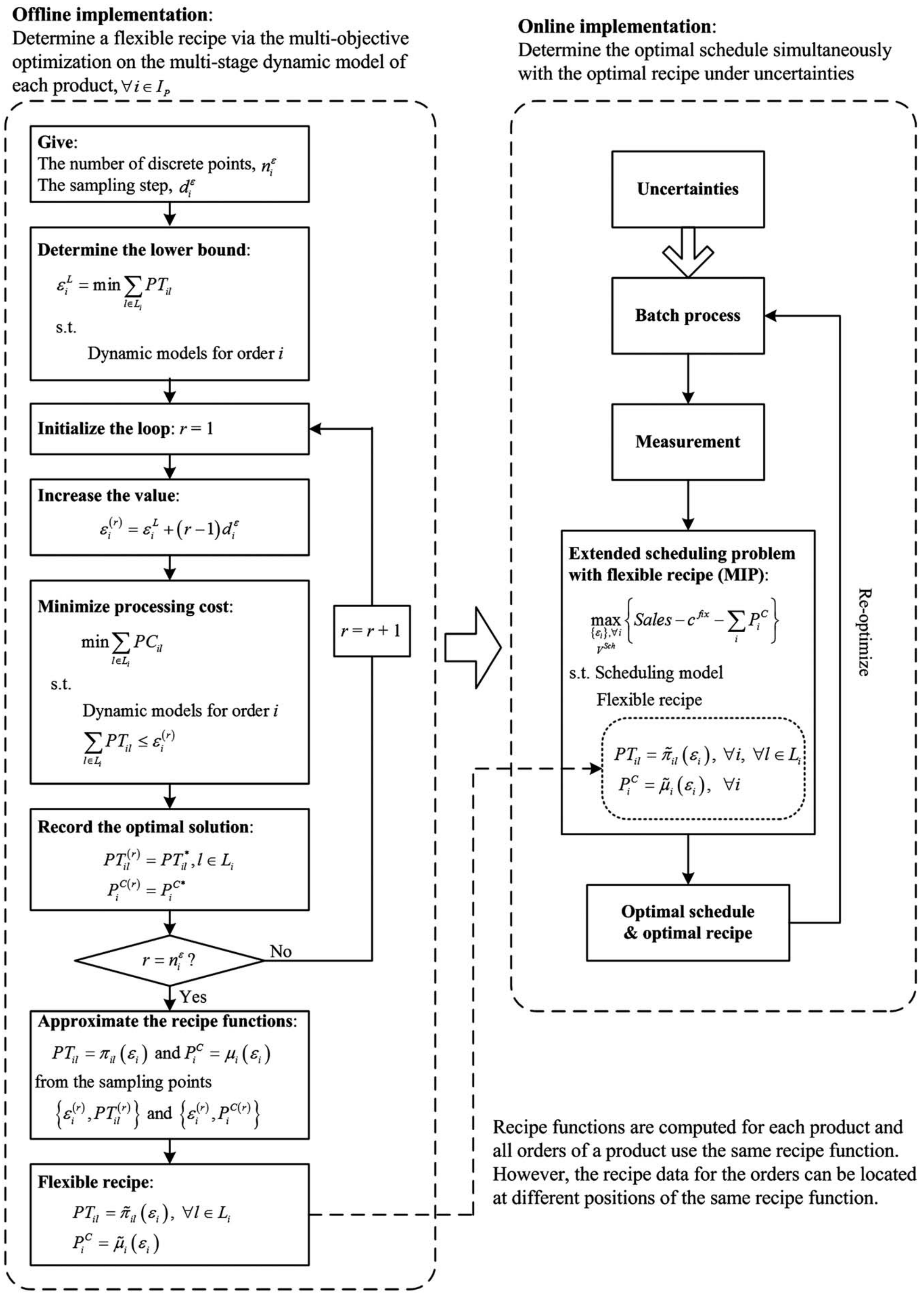

Offline implementation:
Determine a flexible recipe via the multi-objective optimization on the multi-stage dynamic model of each product, $\forall i \in I_P$
Give:
The number of discrete points, n_i^ε
The sampling step, d_i^ε
Determine the lower bound:
$\varepsilon_i^L = \min \sum_{l \in L_i} PT_{il}$ s.t. Dynamic models for order i
Initialize the loop: $r = 1$
Increase the value:
$\varepsilon_i^{(r)} = \varepsilon_i^L + (r-1) d_i^\varepsilon$
Minimize processing cost:
$\min \sum_{l \in L_i} PC_{il}$ s.t. Dynamic models for order i, $\sum_{l \in L_i} PT_{il} \le \varepsilon_i^{(r)}$
Record the optimal solution:
$PT_{il}^{(r)} = PT_{il}^*, l \in L_i$
$P_i^{C(r)} = P_i^{C*}$
$r = n_i^\varepsilon$?
No
Yes
$r = r + 1$
Approximate the recipe functions:
$PT_{il} = \pi_{il}(\varepsilon_i)$ and $P_i^C = \mu_i(\varepsilon_i)$ from the sampling points $\{\varepsilon_i^{(r)}, PT_{il}^{(r)}\}$ and $\{\varepsilon_i^{(r)}, P_i^{C(r)}\}$
Flexible recipe:
$PT_{il} = \tilde{\pi}_{il}(\varepsilon_i), \forall l \in L_i$
$P_i^C = \tilde{\mu}_i(\varepsilon_i)$
Online implementation:
Determine the optimal schedule simultaneously with the optimal recipe under uncertainties
Uncertainties
Batch process
Measurement
Extended scheduling problem with flexible recipe (MIP):
$\max_{\{\varepsilon_i\}, \forall i, V^{Sch}} \left\{ Sales - c^{fix} - \sum_i P_i^C \right\}$
s.t. Scheduling model
Flexible recipe
$PT_{il} = \tilde{\pi}_{il}(\varepsilon_i), \forall i, \forall l \in L_i$
$P_i^C = \tilde{\mu}_i(\varepsilon_i), \forall i$
Optimal schedule & optimal recipe
Re-optimize
Recipe functions are computed for each product and all orders of a product use the same recipe function. However, the recipe data for the orders can be located at different positions of the same recipe function.

Figure 7. Implementation of the integrated method.

recipe functions $PT_{il}=\pi_{il}(\varepsilon_i)$ and $P_i^C=\mu_i(\varepsilon_i)$, while only the extended scheduling problem based on the flexible recipe is required to solve online. The decomposition significantly reduces the computational complexity of the integrated problem, enabling the online application.

The Pareto curve provides a tradeoff between the processing time and the processing cost. The flexible recipe function determined from the Pareto curve records promising operational conditions. The recipe function can be used to encapsulate the information about the dynamic model. Of course, substitution of the dynamic model by the recipe function is an approximation. Due to the complexity of the integrated problem, it is difficult to determine the approximation accuracy for a general scheduling model combined with general dynamic models. However, the proposed method significantly reduces the problem complexity and can be easily implemented. One can simply check its performance for a particular problem from the returned results. As demonstrated in the case study, the solution of the proposed method is very close to the optimal one returned by the simultaneous method. However, the computational time can be reduced by orders of magnitude.

Continuous-time scheduling model based on flexible recipe approximated by piecewise linear function

There are a number of approaches to approximate the recipe function. The approximation method should be integrated with the scheduling model. In this section, we use a piecewise linear function to approximate the recipe function. The approximation is then integrated with the continuous-time scheduling model. After linearization of the bilinear terms, we obtain an MILP problem which is much easier to solve than the original integrated MINLP problem.

After solving the dynamic optimization problems, we obtain recipe data points $\left\{\varepsilon_i^{(r)}, PT_{il}^{(r)}\right\}$ and $\left\{\varepsilon_i^{(r)}, P_i^{C(r)}\right\}$, for each dynamic model. We can create a piecewise linear function by linking the discrete data points with a series of line segments. The approximated functions $PT_{il}=\tilde{\pi}_{il}(\varepsilon_i)$ and $P_i^C=\tilde{\mu}_i(\varepsilon_i)$ can be expressed as

$$\varepsilon_i=\sum_{r\in R_i}\lambda_{ir}\varepsilon_i^{(r)}, \quad \forall i \tag{53}$$

$$PT_{il}=\sum_{r\in R_i}\lambda_{ir}PT_{il}^{(r)}, \quad \forall i, l\in L_i \tag{54}$$

$$P_i^C=\sum_{r\in R_i}\lambda_{ir}P_i^{C(r)}, \quad \forall i \tag{55}$$

$$\sum_{r\in R_i}\lambda_{ir}=1, \quad \forall i \tag{56}$$

$$\lambda_{ir}\geq 0, \quad \forall i, \quad r\in R_i \tag{57}$$

$$\lambda_{i(r=1)}\leq \beta_{i(r=1)}, \quad \forall i \tag{58}$$

$$\lambda_{ir}\leq \beta_{i(r-1)}+\beta_{ir}, \quad \forall i, \quad r\in R_i, \quad 2\leq r\leq |R_i|-1 \tag{59}$$

$$\lambda_{i(r=|R_i|)}\leq \beta_{i(r=|R_i|-1)}, \quad \forall i \tag{60}$$

$$\beta_{ir}\in\{0,1\}, \quad \forall i, \quad r\in R_i \tag{61}$$

The piecewise linear functions are modeled by the convex combination method.[50] Other formulations by the incremental method or the multiple choice method are possible. A piecewise linear function is built for each order i. R_i denotes the index set of the discrete points. Constraints (57)–(61) imply that $\left\{\lambda_{i1},\lambda_{i2},\cdots,\lambda_{i|R_i|}\right\}$ for $\forall i$ is a specially ordered set of Type 2 (SOS2) in which at most two adjacent terms can be nonzero. If a solver supports SOS2 variables, $\left\{\lambda_{i1},\lambda_{i2},\cdots,\lambda_{i|R_i|}\right\}$ can be directly defined as a SOS2 set, which replaces constraints (57)–(61).

When the processing times become variables, the original scheduling model turns out to be nonlinear due to the bilinear terms $W_{ijkl}(st_{il}+PT_{il})$ in Eq. 7. Because the bilinear term is the product of a binary variable and a continuous variable, we can linearize Eq. 7 as

$$TEJ_{jk}=TSJ_{jk}+\sum_{(i,l)\in IL_j}WT_{ijkl}, \quad \forall j, \quad k\in K_j \tag{62}$$

$$WT_{ijkl}+UT_{ijkl}=st_{il}+PT_{il}, \quad \forall i, \quad l\in L_i, \quad j\in J_{il}, \quad k\in K_j \tag{63}$$

$$0\leq WT_{ijkl}\leq W_{ijkl}t_H, \quad \forall i, \quad l\in L_i, \quad j\in J_{il}, \quad k\in K_j \tag{64}$$

$$0\leq UT_{ijkl}\leq \left(1-W_{ijkl}\right)t_H, \quad \forall i, \quad l\in L_i, \quad j\in J_{il}, \quad k\in K_j \tag{65}$$

where nonnegative variables WT_{ijkl} and UT_{ijkl} are introduced for the linearization.

Using the decomposition method and the linearization approach, we transform the integrated MIDO (or MINLP) problem into an MILP scheduling problem based on the flexible recipe, which is summarized as

$$\max_{\substack{\{\varepsilon_i\},\forall i\\ V^{Sch}}}\left\{\text{Sales}-c^{fix}-\sum_i P_i^C\right\} \tag{66}$$

s.t.

Scheduling model (Eqs. 1–6, 62–65, 8–18)
Recipe data (Eqs. 53–61)

where the scheduling model (1)–(6) is the time-slot model.[35]

Discrete-time scheduling model based on flexible recipe approximated by discrete points

Another approach to approximate the recipe function uses a set of discrete points. The approximation can be integrated with the discrete-time scheduling model, in which the time interval of a processing unit is partitioned into a set of time points with the step size denoted by d_t. For consistency, we set the step size of the discrete-time scheduling model as the step size used in the approximation of the recipe function, that is, $d_t=d_i^{\varepsilon}$.

In the discrete-time scheduling model, the processing times should be equal to an integer times the step size

$$PT_{il}^{(r)}=\left\lceil\frac{PT_{il}^L}{d_t}\right\rceil d_t+(r-1)d_t, \quad \forall i, \quad l\in L_i \tag{67}$$

where PT_{il}^L denotes the lower bound of PT_{il}. Appending the constraint (67) into the optimization problem (47), we obtain the recipe data of $\left\{\varepsilon_i^{(r)}, PT_{il}^{(r)}\right\}$ and $\left\{\varepsilon_i^{(r)}, P_i^{C(r)}\right\}$. The processing times in the discrete-time scheduling model is represented as

$$DPT_{ilr}=PT_{il}^{(r)}/d_t+\lceil st_{il}/d_t\rceil \tag{68}$$

The ratio $PT_{il}^{(r)}/d_t$ is an integer due to constraint (67).

The discrete-time scheduling model is modified from the one for the batch process with a network structure.[51,52] In

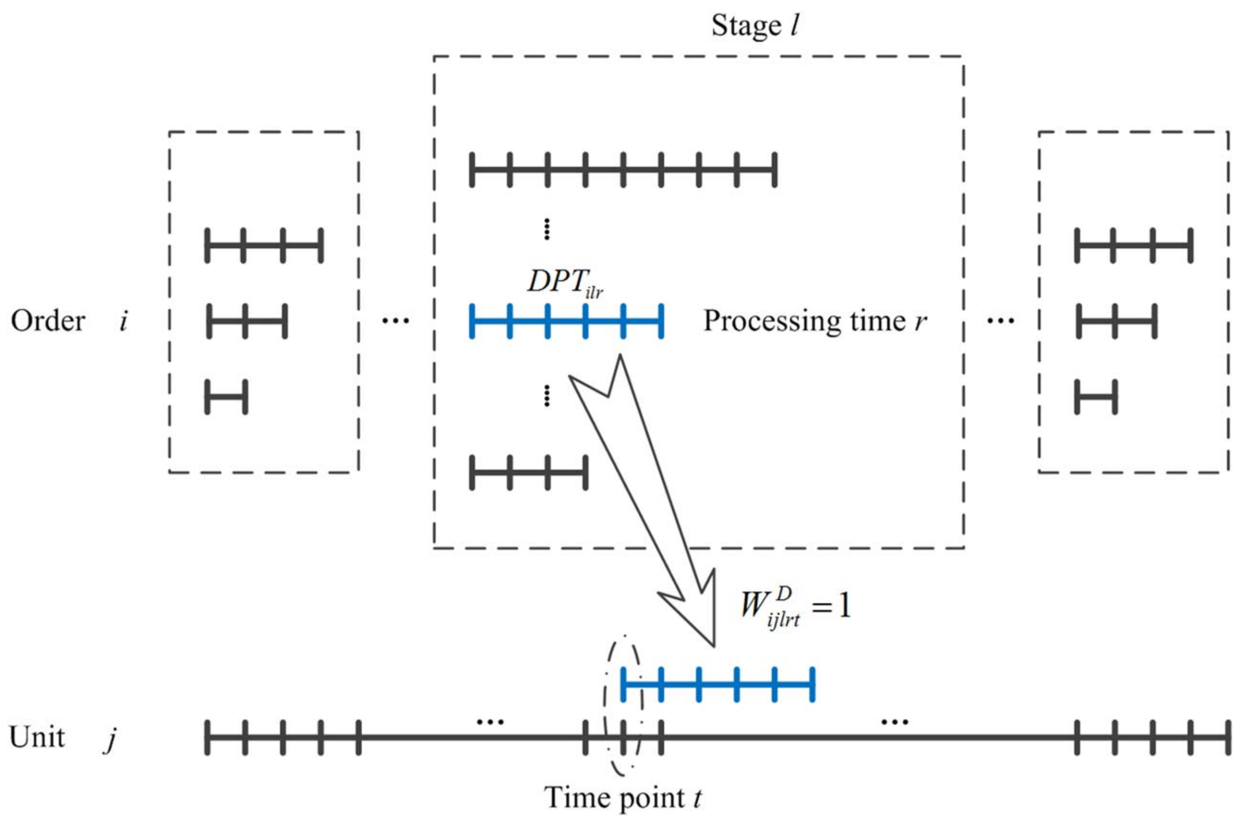


Figure 8. Illustration of the allocation variable W^D_{ijlrt} in the discrete-time scheduling model.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

the discrete-time scheduling model, the time point index is denoted by t. A new allocation variable W^D_{ijlrt} is introduced, which is the counterpart of the allocation variable W_{ijkl} in the continuous-time model. If $W^D_{ijlrt}=1$, stage l of order i is executed in unit j starting from the time point t and using the processing time r. The allocation by using W^D_{ijlrt} is illustrated in Figure 8. The allocation variables are subject to the constraint of

$$\sum_{j\in J_{il}}\sum_{r\in R_i}\sum_{t}W^D_{ijlrt}=1,\quad \forall i,\quad j\in L_i \tag{69}$$

The constraint ensures that every stage of an order is executed only once. When a unit is assigned to a task, it cannot be used by others until the task is finished. The constraint on the unit availability is expressed as

$$\sum_{i}\sum_{l\in L_i}\sum_{r\in R_i}\sum_{t'=t-DPT_{ilr}+1}^{t}W^D_{ijlrt'}\le 1,\quad \forall j,\quad t \tag{70}$$

For the sequential process, an order stage can be executed only after the previous stage finishes. The constraint on the stage sequence is given by

$$\sum_{j\in J_{il}}\sum_{r\in R_i}\sum_{t'\le t}W^D_{ijlrt'}\le\sum_{j\in J_{il}}\sum_{r\in R_i}\sum_{t'\le t-DPT_{i(l-1)r}}W^D_{ij(l-1)rt'},\quad \forall i,\quad t,\quad l\in L_i\quad\text{and}\quad l\ge 2 \tag{71}$$

The processing time index r is only dependent on the order index i because the processing times for different stages are calculated from the same solution of the optimization problem (50). This condition implies that the stages of an order should select the processing times with the identical index r, resulting in the equalities on the allocation variables

$$\sum_{j\in J_{il}}\sum_{t}W^D_{ijlrt}=\sum_{j\in J_{i(l-1)}}\sum_{t}W^D_{ij(l-1)rt},\quad \forall i,\quad r\in R_i,\quad l\in L_i\quad\text{and}\quad l\ge 2 \tag{72}$$

In the discrete-time scheduling model, all the timing relations are expressed by the constraints on the allocation variables and there is no need to express the starting times and the ending times of an order stage explicitly. The ODT is modeled by

$$ODT_i\ge\sum_{j\in J_{il}}\sum_{r\in R_i}W^D_{ijlrt}(t+DPT_{ilr})d_t,\quad \forall i,\quad t,\quad l=|L_i| \tag{73}$$

where t on the right-hand side is the index of the time points. The processing cost is given by

$$P^C_i=\sum_{l\in L_i}\sum_{j\in J_{il}}\sum_{r\in R_i}\sum_{t}W^D_{ijlrt}P^{C(r)}_i,\quad \forall i \tag{74}$$

The discrete-time scheduling model based on the flexible recipe is summarized as follows

$$\max_{\substack{\{\varepsilon_i\},\forall i\\ V^{Sch}}}\left\{\text{Sales}-c^{fix}-\sum_i P^C_i\right\} \tag{75}$$

s.t.

Scheduling model (Eqs. 69–73, 12–19)
Processing time and processing cost (Eqs. 68 and 74)

The discrete-time scheduling model involves more binary variables than the continuous-time scheduling model. However, the structure of the discrete-time model is simpler than

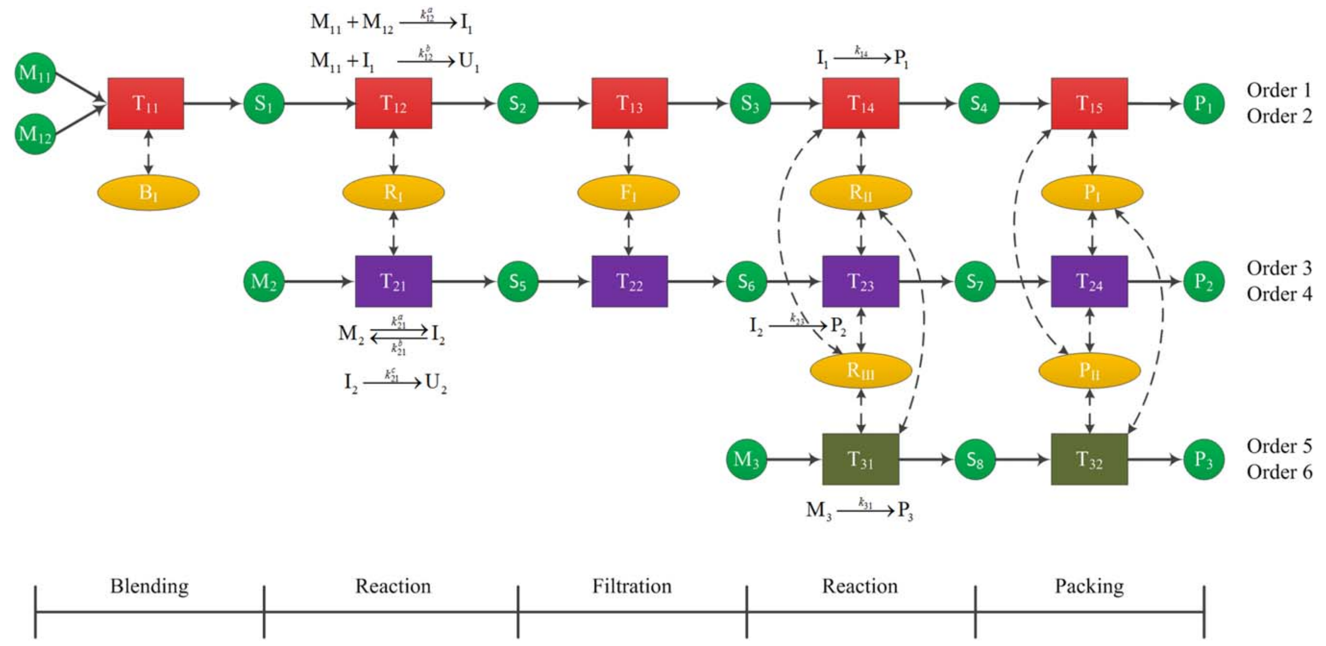

$M_{11}+M_{12} \xrightarrow{k_{12}^{a}} I_1$
$M_{11}+I_1 \xrightarrow{k_{12}^{b}} U_1$
$I_1 \xrightarrow{k_{14}} P_1$
$M_2 \underset{k_{21}^{b}}{\overset{k_{21}^{a}}{\rightleftarrows}} I_2$
$I_2 \xrightarrow{k_{21}^{c}} U_2$
$I_2 \xrightarrow{k_{23}} P_2$
$M_3 \xrightarrow{k_{31}} P_3$
Order 1
Order 2
Order 3
Order 4
Order 5
Order 6
Blending
Reaction
Filtration
Reaction
Packing

Figure 9. The RTN representation of the flexible jobshop scheduling problem.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

that of the continuous-time model. Thus, the better choice of the two scheduling models is usually problem-dependent. We implement both models and compare them based on the results.

Case Study

The proposed integration framework is applied to a flexible jobshop scheduling problem in this section. We first present the problem description including the RTN representation of the scheduling problem and the differential equations of the dynamic models. Then, we solve the multiobjective, multistage dynamic optimization problems independently for each production line containing different numbers of operational stages. The core component of the proposed method is the expression of the information associated with the dynamic models by the Pareto frontiers. To demonstrate the advantages of the proposed method, we compare it with the sequential method and the simultaneous method. The efficiency of the proposed method enables its online application which is critical to dealing with uncertainties. We consider a simulated example of the equipment breakdown to illustrate the online application.

In this work, all optimization problems are modeled in GAMS 23.8.1 and solved in a PC with Intel(R) Core(TM) i5-2400 CPU @ 3.10 GHz, 8-GB RAM, and Window 7 64-bit operating system. Though the CPU has four cores, only one core is used in the computation.

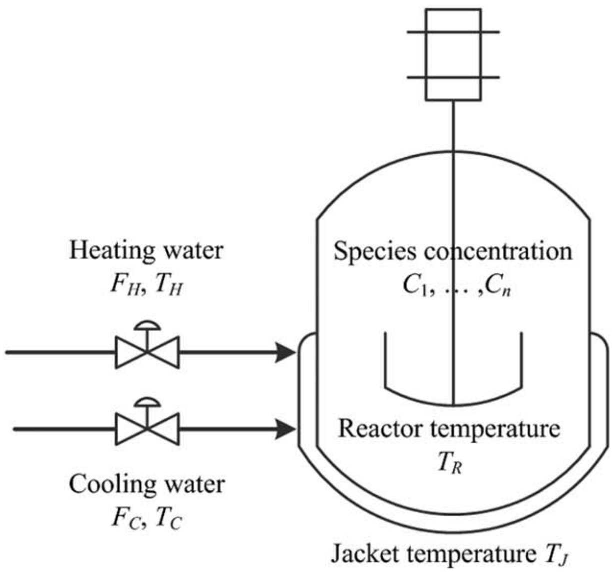
Problem description

The RTN representation of the flexible jobshop scheduling problem is displayed in Figure 9. The sequential process contains three production lines which have different operational stages. The first stage for product P_1 is the blending task which mixes the two reactants M_{11} and M_{12}. The second stage is a reaction task containing multiple reactions. The third stage is a filtration task which filters out the undesired byproduct. The fourth stage is a first-order reaction task. The fifth stage is the packaging task executed in a packaging line. The feature of parallel units is taken into account in this model. Reactor R_{II} is identical to Reactor R_{III} while Packaging line P_I is identical to P_{II}. Thus, the reaction in the fourth stage of P_1 can take place in either R_{II} or R_{III}. Similarly, the packaging task can be executed in either packaging line. The production line for P_2 is similar to that for P_1 except the blending task. The production line for P_3 only contains a first-order reaction task in the first stage and the packaging task in the last stage.

In this model, the dynamics of the reaction tasks is important. Each reaction task is described by a dynamic model. However, the blending task for P_1, the filtration tasks for P_1 and P_2, and the packaging tasks for P_1, P_2, and P_3 are assumed to have fixed recipe and can be described with algebraic models. A general structure of the dynamic models for the reaction tasks is displayed in Figure 10.

The reaction task takes place in a batch reactor with a jacket. The differential equations describe the chemical kinetics, the reactor temperature, and the jacket temperature. The chemical kinetics follows the mass rate law where the reaction order of a species is determined by its stoichiometric coefficient in the corresponding reaction. The rate coefficient is expressed according to the Arrhenius equation. The control variables are the flow rate of the heating water and the flow rate of the cooling water in the jacket. Manipulating these flow rates can change the jacket temperature, then the reactor temperature, and finally the chemical kinetics by altering the reaction rates.

The objective is to maximize the production profit which is equal to the product sales minus the production cost. The decision variables include: (1) the scheduling decisions, for example, the task sequences and the equipment assignments; (2) the operating recipe for the reaction tasks, for example, the heating/cooling water flow rates, the temperature profiles, and the processing times. The data for this flexible jobshop scheduling model are all listed in the Appendix.



$$\begin{cases} \dfrac{d}{dt}C_1(t) = \sum_{i=1}^{n} \nu_{1i} r_i \\ \vdots \\ \dfrac{d}{dt}C_m(t) = \sum_{i=1}^{n} \nu_{mi} r_i \\ \dfrac{d}{dt}T_R(t) = \dfrac{\sum_{i=1}^{n} r_i(-\Delta H_i)}{\rho_R c_R} + \dfrac{UA(T_J - T_R)}{v_R \rho_R c_R} \\ \dfrac{d}{dt}T_J(t) = \dfrac{F_H T_H + F_C T_C - (F_H + F_C)T_J}{V_J} + \dfrac{UA(T_R - T_J)}{V_J \rho_J c_J} \end{cases}$$

$$r_i = k_i \cdot \prod_{j=1}^{m} (C_j)^{\nu_{ij}}$$

$$k_i = Z_i e^{-E_i/T_R}$$

Figure 10. The dynamic model of a reaction task.

Multistage dynamic optimization

The first step of the proposed method is to solve the dynamic optimization problem for each production line. This subsection uses the production line of P_1 to illustrate the optimization procedure. The dynamic optimization problems for other products P_2 and P_3 are solved in the same way. As shown in Figure 9, a production line contains the stages with a fixed recipe besides the stages of reaction tasks. Because we do not consider dynamic optimization for tasks with the fixed recipe, these stages are removed from the optimization approach. For example, in the production line of P_1, only the second and the fourth stages need to be optimized by solving the dynamic optimization problems.

Figure 11 displays the results of the dynamic optimization. The initial concentration of I_1 in the fourth stage is equal to the final concentration of I_1 in the second stage. However, the initial value of the reactor temperature in the fourth stage is set as the ambient temperature (300 K) due to the heat loss in material transition. In each reaction task, the reactor temperature is increased by the heating water in the beginning time interval. A high-temperature results in a fast reaction rate. The reactor temperature cannot, however, exceed the safety boundary. The maximum temperate of reactor R_I which executes the reaction task in the second stage is set as 350 K and the maximum temperate of reactor R_{II} or R_{III} which execute the reaction task in the fourth stage is set as 360 K. Because the reactions can be exothermic, it is possible that the jacket temperature exceeds the temperature of the heating water and the upper bound on the jacket temperature is set as 370 K for all reactors.

At the end of the reaction task, the reactor temperature needs to be decreased by the cooling water because the materials with the high temperature cannot be processed by the following stage. The outlet temperature should be less than 320 K. The utility constraints are imposed on the sum of the heating water flow rate and the cooling water flow rate. The upper bound on the sum of water flow rates is 20 m^3/h for reactor R_I in the second stage, whereas the upper bound is 30 m^3/h for reactors R_{II} and R_{III} in the fourth stage. The effects caused by those constraints are reflected on the dynamic trajectories in Figure 11. The output of the fourth stage is the final product. The final concentration of P_1 should not be less than 4000 kmol/m^3 to satisfy the quality demand. The data about the dynamic models are listed in the Appendix.

For the multiobjective optimization, we calculate the lower bound of the total processing time to determine the starting point of the Pareto curve. The lower bound is obtained by minimizing the total processing time of the two stages. The optimal value is $0.85+1.45=2.30$ h. The decision variables are the heating flow rates and the cooling flow rates, which are expressed as a piecewise constant function with 10 intervals. The collocation method[30] is applied to the dynamic optimization. The method discretizes the continuous-time variables in the dynamic models by finite elements and collocation points. There are two dynamic models, one for each stage. There are 20 finite elements in each model, and three collocation points in every finite element. The length of a finite element in each dynamic model is treated as a variable to be optimized. Therefore, the processing time of a task can be changed. After discretization, the dynamic optimization problem is transformed into an NLP problem. The model and solution statistics are listed in Table 2.

The dynamic optimization problem is easy to solve and the optimal solution is found within 1.2 s. The same procedure is applied to other production lines.

Proposed integration method with multiobjective, multistage dynamic optimization

The second step of the proposed method is the multiobjective dynamic optimization, which reveals the conflicting factors inside the scheduling problem. Balancing these conflicting factors is the motivation of considering an integrated approach. The conflicting factors are the processing time and the processing cost. The processing cost is mainly due to the consumption of heating water and cooling water.

The multiobjective optimization is performed for each production line independently. The Pareto curve is generated by using the ε-constraint method. A set of incremental processing times is generated starting from the minimum processing time calculated in the previous section. The time step between two adjacent values is 0.1 h and totally 10 processing times are generated in a set. Then at each processing

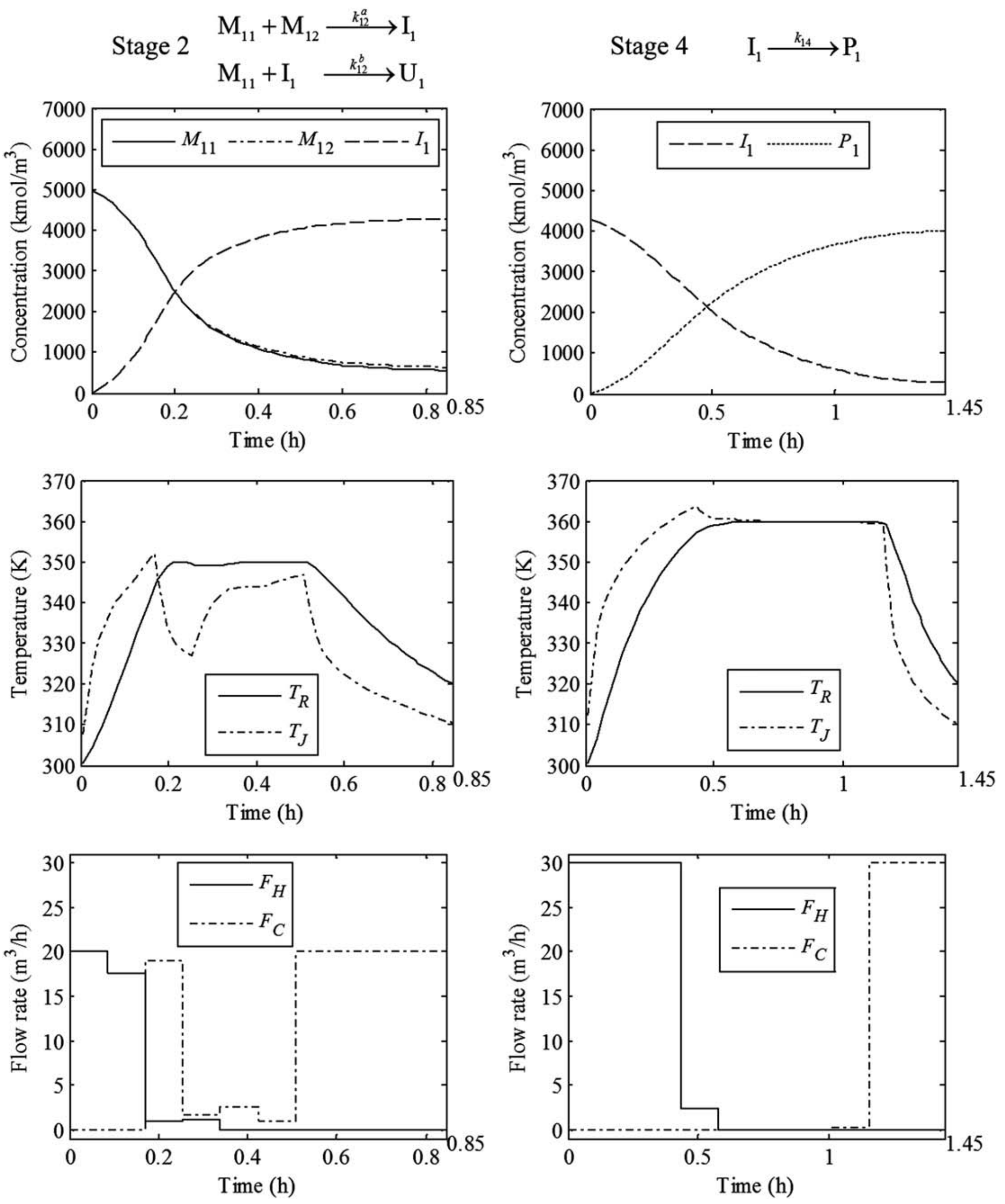


Figure 11. Multistage optimization on the second and the fourth stage of P_1. The two stages are optimized simultaneously and the total processing time of the two stages is minimized.

time, the processing cost is minimized. An approximated Pareto curve is generated by linking the points of total processing time and total processing cost with straight lines. The multiobjective optimization problem is solved to all production lines for P_1, P_2, and P_3. For the products P_1 and P_2, the processing time or the processing cost is the total value of the two reaction stages, that is, the multistage optimization is performed to generate the points on the Pareto curve.

Table 2. Model and Solution Statistics for the Multistage Dynamic Optimization of P_1

Model	Finite Elements	Collocation Points per Finite Element	Equations	Variables
	20	3	1818	1791
Solution	Type	Solver	Objective (h)	CPU (s)
	NLP	CONOPT3	2.30	1.2

The objective is to minimize the total processing time of the two reaction stages.

The resulting Pareto curves are displayed in Figure 12. It is seen from the Pareto curve that the processing cost strictly decreases with the processing time for each product. The dynamic models are for the reactions. A large processing time indicates a small reaction rate, which in turn implies a small consumption of heating/cooling water and consequently a lower-processing cost. On the contrary, to reduce the processing time, more control efforts should be placed on the process to manipulate the reaction rate. More heating/cooling water is consumed, implying a high-processing cost.

Because the processing cost increases as the processing time decreases, we cannot reduce both variables simultaneously. They are conflicting factors. The points on the Pareto curves determine the recipe candidates, which replace the

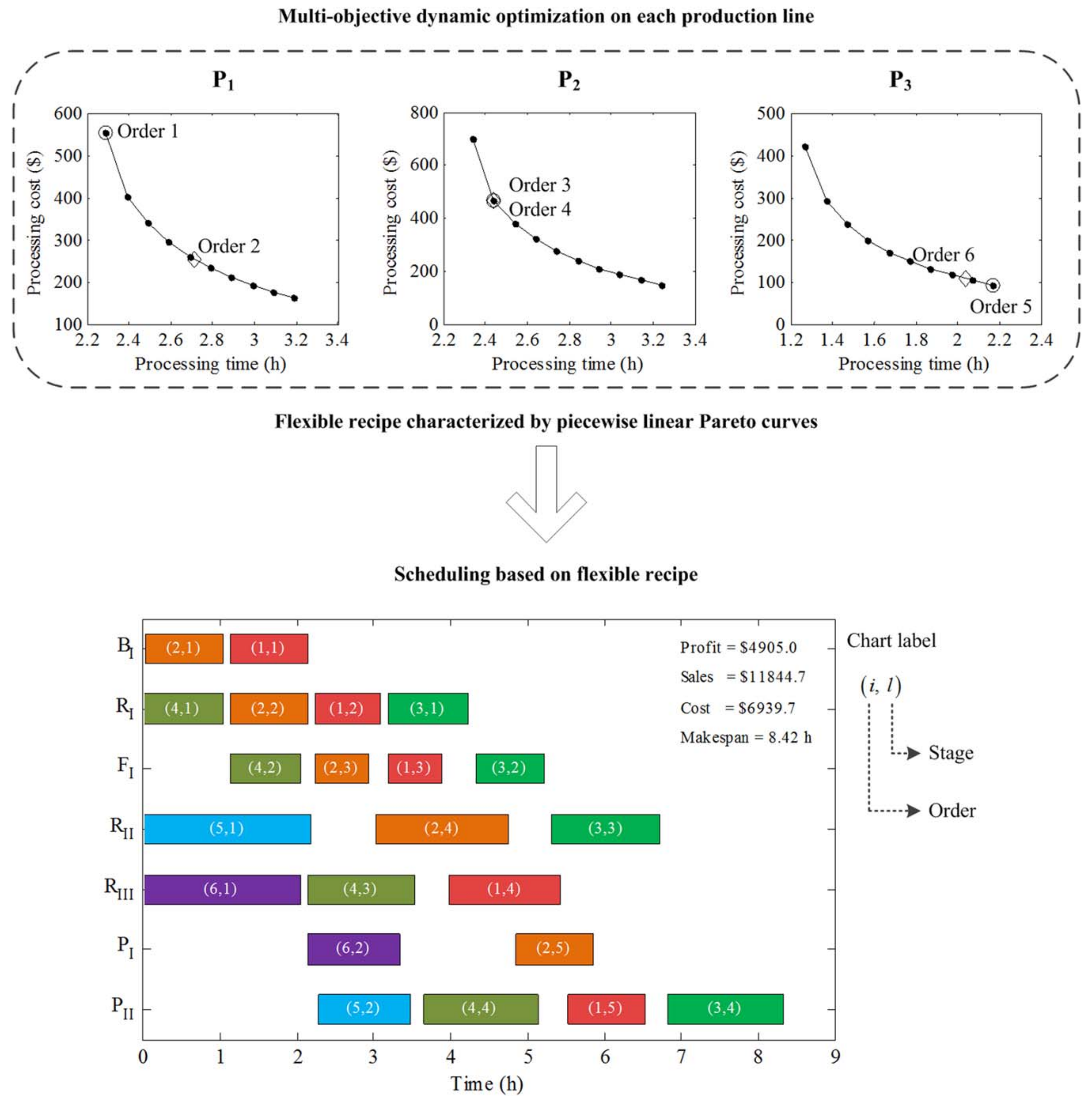


Figure 12. Scheduling returned by the proposed method based on the piecewise linear Pareto curves and the continuous-time scheduling model. The optimal operating conditions are located in the Pareto curves.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

dynamic models in the integrated problem. Based on the piecewise linear Pareto curve, the continuous-time scheduling model is used to formulate the integrated problem. The resulting integrated problem is an MILP problem and the optimal scheduling decisions are determined simultaneously with the optimal recipe characterized by the Pareto curves. The model and solution statistics are listed in Table 3 (the second column) and the Gantt chart is displayed in Figure 12.

The optimal operating conditions denoted by the optimal recipe are located back on the Pareto curves. The locations of the operating conditions selected by the integrated method vary for different products as well as different orders. Order 1 and Order 2 are associated with the same product P_1. However, their operating conditions are different. The operating condition of Order 1 is located on the left end of the Pareto curve where the processing time reaches the minimum value. The operating condition of Order 2 is located around the middle point of the Pareto curve.

Even for the same product, the locations of the operating conditions can be different with respect to orders because the orders are executed in different production environments. It is a feature of the integrated method which optimizes the recipe according to the scheduling decisions. The widely-spread operating points on the Pareto curves indicate that the conflicting factors are balanced in different ways according to the products and the orders. It is difficult to make a good tradeoff

Table 3. Model and Solution Statistics for Different Methods

Method	**Proposed** (Continuous-Time)	**Proposed** (Discrete-Time)	Sequential[a]	Simultaneous[b]
Recipe	Optimized	Optimized	Fixed	Optimized
Type	MILP	MILP	MILP	MINLP
Constraints	4548	2213	614	9262
Con. Var.	3973	2912	185	9145
Dis. Var.	282	12,120	186	186
Solver	CPLEX 12.4	CPLEX 12.4	CPLEX 12.4	SBB
Objective ($)	4905.0	4693.7	3776.3	5074.9
CPU (s)	15.6	19.7	1.2	50,000
Gap (%)	0	0	0	45.7

The objective is to maximize the production profit.
[a]The model for the sequential method represents the scheduling problem where the recipe is fixed.
[b]For the simultaneous method, there are 20,016 nodes searched in the branch and bound procedure. The optimal solution is found at 19,598 nodes when the computational time is equal to 49,133 s.

without knowledge of the scheduling problem. This is the reason that the scheduling and dynamic optimization problems should be solved simultaneously.

The Pareto curve can also be expressed by the discretized points. This expression is integrated with the discrete-time scheduling model. The scheduling results returned by the proposed method based on the discretized Pareto points and the discrete-time scheduling model is shown in Figure 13. The time step of the discrete-time scheduling model is equal to the step used to obtain the Pareto points (0.1 h). It should be noted that there are additional constraints added in the multiobjective optimization: The processing time of each stage should be an integer times the time step so that it can be used without rounding errors in the discrete-time scheduling model. In contrast, in the previous case of the piecewise linear Pareto curve, the processing time of each stage can be a fractional value times the time step. Due to the constraints (67), the Pareto points on Figure 13 might not be the same as the discrete points on the Pareto curves in Figure 12.

The model and solution statistics are listed in Table 3 (the third column). The integrated problem based on the discrete-time model contains many more binary variables than the problem based on the continuous-time model. However, the discrete-time model has a much simpler structure than the continuous-time model. For example, there are no big-M constraints for time slot matching in the discrete-time model. Consequently, the computational time of the discrete-time model is only slightly longer than that of the continuous-time model (19.7 vs. 15.6 s). Because the processing time of each stage is restricted in the discrete-time model, the optimal value of the profit is slightly smaller (by 4.3%) than that returned by the continuous-time model ($4693.7 vs. $4905.0).

Comparisons with sequential method and simultaneous method

To demonstrate the performance of the proposed method, we compare it with the sequential method and the simultaneous method in this section. The parameter values, which are equal to those used in the previous section, are given in the Appendix. The differences among the two integrated methods and the sequential method are visualized in Figure 14. Both integrated methods based on the flexible recipe, but the recipe is expressed in a different way. Detailed discussions about the differences and the comparisons are presented in the following.

Conventionally, scheduling is based on the fixed recipe determined by dynamic optimization. This is the sequential method because the dynamic optimization and the scheduling optimization are solved one after the other. Being solved beforehand, the dynamic optimization cannot use the objective function of the scheduling problem. Thus, it has to be performed based on some local criterion. We use the total processing time for each production line as the objective function. This criterion attempts to finish an order execution through all stages as quickly as possible to avoid the penalty on the delayed delivery date.

After the dynamic optimization on each production line, the processing times and the processing costs are calculated. These variables are then fixed and actually become the parameters for the sequential scheduling problem. If a product has multiple orders, all orders use the same recipe. The sequential scheduling problem is then solved based on the fixed recipe returned by the dynamic optimization. The scheduling results are displayed in Figure 15. The model and solution statistics of the sequential scheduling problem is listed in Table 3 (the fourth column). Each dynamic model is discretized by 20 finite elements and each finite element has three collocation points. The sequential scheduling problem is easy to solve because the processing times and the processing costs all become parameters. The computational time of the sequential scheduling problem is less than that of the proposed method (1.2 vs. 15.6 s) by using the continuous-time formulation for both integrated scheduling problems.

The makespan of the sequential method is shorter than that of the integrated method (7.69 vs. 8.42 h) because the total processing time of each product is minimized. The shorter makespan contributes to a larger order price, reflected by a larger value of the sales ($12,122.1 vs. $11,844.7). However, the cost is larger ($8345.7 vs. $6939.7) caused by the small processing times. It is due to the inability of making a good balance between the conflicting factors that the sequential method results in a lower profit than that of the integrated method ($3776.3 vs. $4905.0).

To overcome the drawback of the sequential method, integrated methods are proposed. A common one is the simultaneous method,[33,34] which solves the scheduling problem with all dynamic optimization models simultaneously. The dynamic models are discretized and enter the scheduling problem as a large set of nonlinear equations. For consistency, we use the same parameters to discretize the dynamic models, that is, 20 finite elements for each model and three collocation points for each element.

For the simultaneous method, the number of dynamic models in the integrated problem increases as the number of orders. There are two dynamic models for P_1, two models for P_2, and one model for P_3. Each product has two orders so

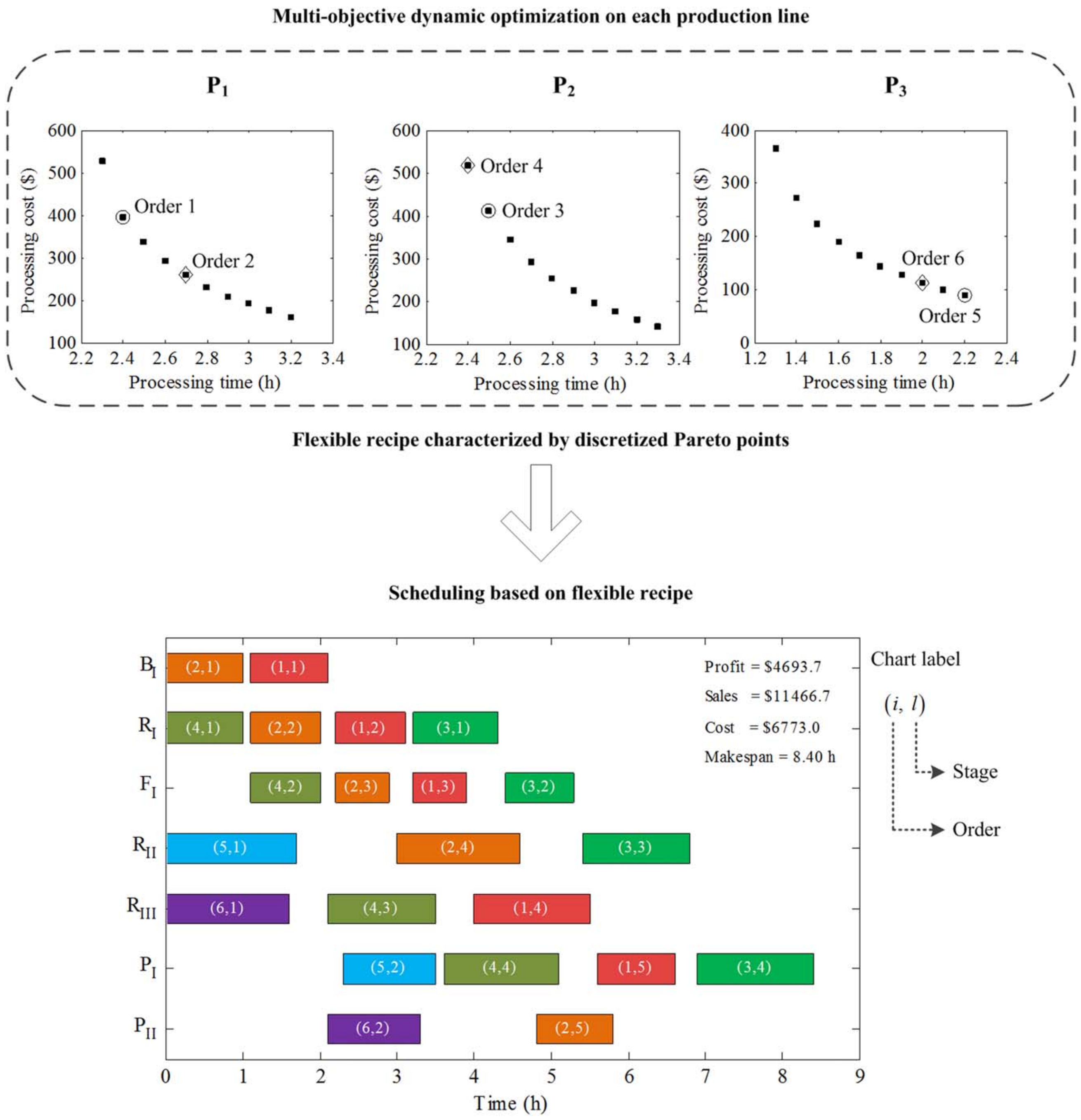


Figure 13. Scheduling returned by the proposed method based on the discretized Pareto points and the discrete-time scheduling model; The optimal operating conditions are located in the selected Pareto points.
[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

there are totally 2 × (2+2+1)=10 dynamic models in the integrated problem. If we place more orders for the products, more dynamic models should be included in the integrated problem. In contrast, the sequential method and the proposed method solve the dynamic optimization problem for each product. The number of dynamic models required to be optimized is independent on the number of orders.

The model and solution statistics are listed in Table 3 (the fifth column). Even though the scheduling problem is linear, the integrated problem turns out to be highly nonlinear after we include the large set of equations resulting from discretizing the dynamic models. The MINLP problem is solved by SBB, which is commonly used in literature about integrating scheduling and dynamic optimization.[25,33,34,53] The MINLP problem is initialized with the solution of the sequential method, including variables in the scheduling problem and variables in all dynamic models. If the integrated problem is not initialized properly, the problem might be local infeasible.

Owing to the computational complexity, there is still a large gap of 45.7% when the resource limit of 50,000 s is reached. The scheduling results are shown in Figure 16a. When we initialize the problem at some other randomly selected conditions, solutions with different gaps are returned. In all cases, the gaps are large (more than 40%) when the solver stops at the resource limit of 50,000 s.

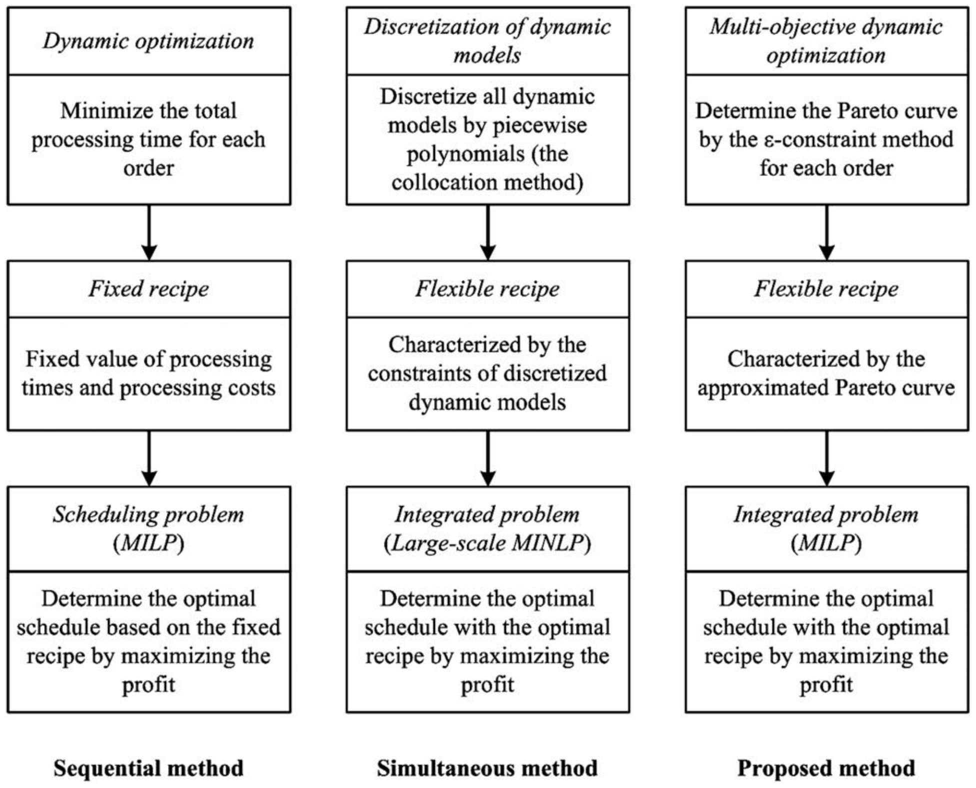

Dynamic optimization
Minimize the total processing time for each order
Fixed recipe
Fixed value of processing times and processing costs
Scheduling problem (MILP)
Determine the optimal schedule based on the fixed recipe by maximizing the profit
Discretization of dynamic models
Discretize all dynamic models by piecewise polynomials (the collocation method)
Flexible recipe
Characterized by the constraints of discretized dynamic models
Integrated problem (Large-scale MINLP)
Determine the optimal schedule with the optimal recipe by maximizing the profit
Multi-objective dynamic optimization
Determine the Pareto curve by the ε-constraint method for each order
Flexible recipe
Characterized by the approximated Pareto curve
Integrated problem (MILP)
Determine the optimal schedule with the optimal recipe by maximizing the profit
Sequential method
Simultaneous method
Proposed method

Figure 14. Comparisons of different methods solving the scheduling problem and the dynamic optimization problems.

The resulting optimal profit is \$5074.9 and it is higher than \$3776.3 of the sequential method. The increase in the profit exemplifies the superiority of integrating both scheduling and dynamic optimization problems. However, the computational time is much longer, which is about 13.9 h. The computational time is even larger than the scheduling horizon of 9 h. The profit is slightly larger, by 3.3%, than that returned by the proposed method based on the continuous-time scheduling model. However, the computational time is more than three orders of magnitude longer. The inefficiency in solving the MINLP problem has also been reported in some other examples.[33,34]

To further analyze the computational bottleneck of the simultaneous approach, we solve the problem again by fixing the binary assignment variables at those returned by the proposed method based on the continuous-time scheduling model. The assignment variables are the main binary variables in the model. The scheduling results are displayed in Figure 16b. The profit is now slightly higher than the previous solution (\$5234.6 vs. \$5074.9). The computational time

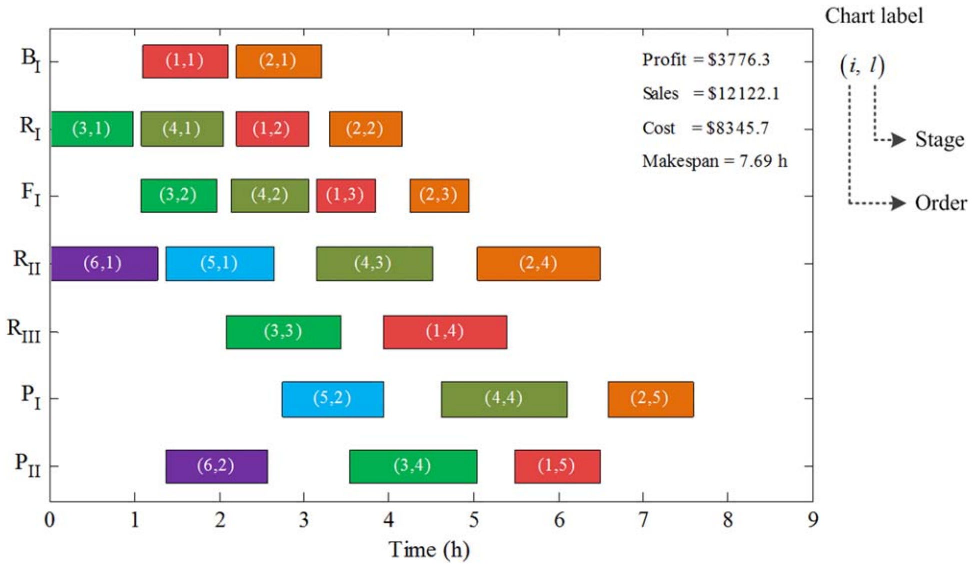
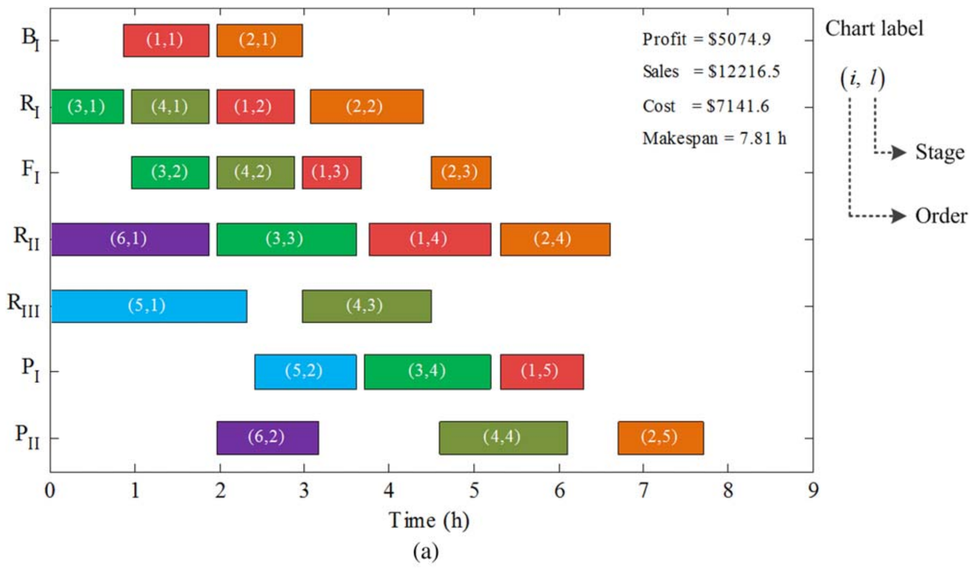


Figure 15. Scheduling results returned by the sequential method; The schedule is determined based on the fixed recipe returned by the multistage dynamic optimization which minimizes the total processing time of each production line.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]



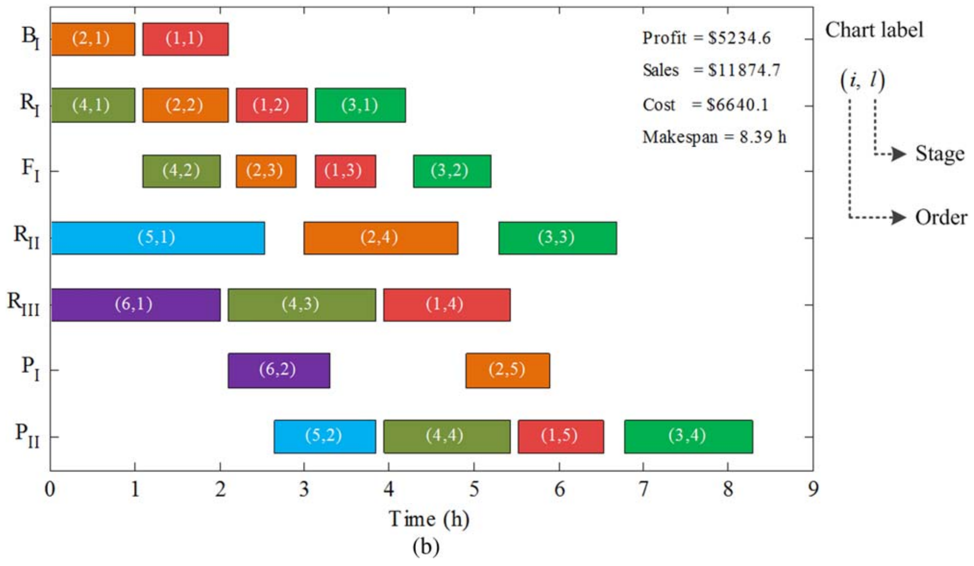


Figure 16. Scheduling results returned by the simultaneous method via: (a) initialization from the sequential method, and (b) initialization from the proposed method and fixing assignment variables to the value returned by the proposed method.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

is 76.3 s to reach zero gap. However, because good values of the binary variables cannot be identified in advance, such a good solution with a short computational time is difficult to obtain in practice. It is also seen that the dynamic optimization is straightforward even if the dynamic models for all production lines and all orders are solved together. However, when the dynamic models are combined with the discrete scheduling decisions, the highly nonlinear equations enter the scheduling problem, turning the original MILP problem to be a complicated MINLP problem with a much larger size.

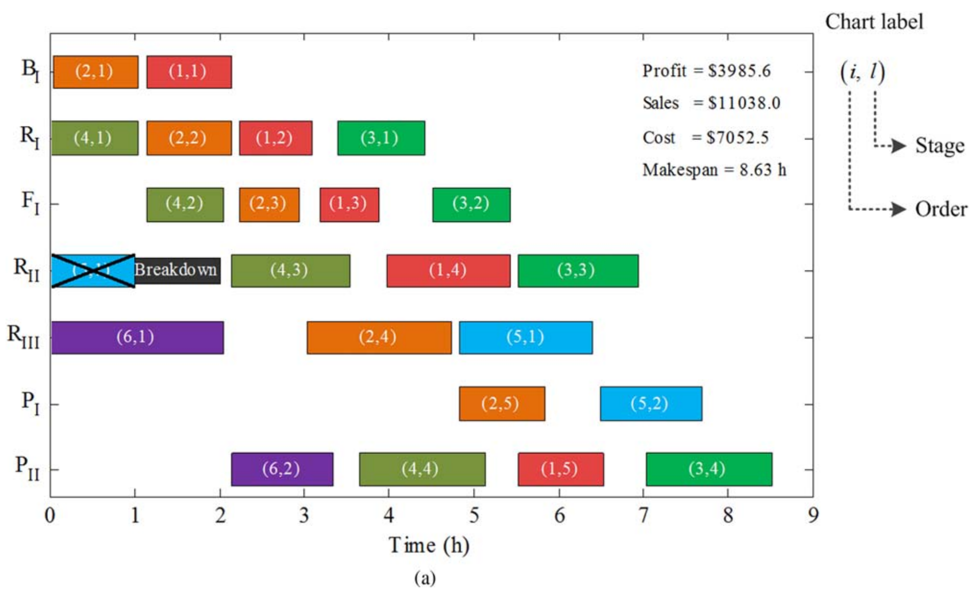
Online implementation under uncertainties

Batch production is subject to different types of uncertainties. Under uncertainties, an optimal schedule generated offline might not remain optimal or might even become infeasible. Thus, online rescheduling is often required. For online application, computational time is critical and the schedule has to be determined in a short time period.

The proposed integrated method decomposes all dynamic models from the scheduling model. Only an MILP scheduling problem with a flexible recipe is required to be solved online. The efficiency of the proposed method enables its online application. For demonstration, we solve the integrated problem under the uncertainty of equipment breakdown. Other uncertainties, for example, variations in processing times and changes in orders, can be dealt with in the same way.

As illustrated in Figure 7, we implement the method online in a reactive way. The uncertainties are not taken into account in the scheduling model. However, when they occur, we obtain the measurement and resolve the integrated problem according to the updated information. The returned new schedule for the unfulfilled orders and new operational recipe are fed back to the process.

The process initially operates according to the offline schedule (Figure 12). At Hour 1, reactor R_{II} breaks down. The breakdown interrupts the execution of Stage 1 for Order 5. The batch needs to be reproduced. The reactor recovers at Hour 2



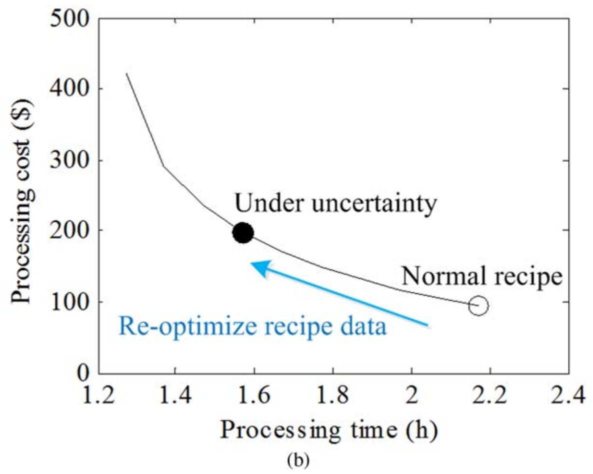


Figure 17. Scheduling results under equipment breakdown; (a) online rescheduling; (b) reoptimized recipe data for Order 5.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

after repairing. The uncertainty of the equipment breakdown disturbs the predetermined schedule and the production needs to be rescheduled. Rescheduling is conducted online. When the reactor breaks down or when it is recovered, the integrated problem is solved again. The decision variables in the past horizon are fixed at the historical data. The scheduling results are shown in Figure 17a. The sequence of future task executions is changed from that in the offline schedule. After reactor R_{II} is recovered, the task of Stage 3 for Order 4 is executed. This task was assigned to the parallel reactor R_{III} in the offline schedule. The two stages for Order 5 are postponed to the end of the schedule until other orders are processed.

Not only the scheduling decisions but also the recipe are changed in the rescheduling. The conflicting factors of processing times and processing costs are balanced in a new way according to the uncertainty. A new recipe data for Order 5 is determined and displayed in Figure 17b. To reduce the loss in the delivery dates delayed by the breakdown, more weights are placed on the processing times. The processing time decreases while the processing cost increases. This example demonstrates that the recipe should be optimized simultaneously with scheduling decisions. The proposed method can reoptimize them in an integrated way online under uncertainties.

Conclusions

We proposed a novel framework to solve the integrated problem of scheduling and dynamic optimization for sequential batch processes. The integrated model was built by extending the time-slot scheduling model and the order price function. The number of dynamic models in the integrated problem was not dependent on the time points. To reduce the computational time for online application, we developed a decomposition method. All dynamic models were decomposed from the scheduling model, and information about the dynamic models was characterized by a flexible recipe. The recipe data were determined by the Pareto frontiers of the processing time against the processing cost, which were generated by the multiobjective dynamic optimization on each product. All dynamic optimization problems were solved offline, independently from each other and separately from the scheduling problem. After the decomposition procedure, the integrated problem was approximated by a scheduling problem with recipe optimization. This is an MILP problem and much easier to solve than the complicated MINLP problem without decomposition.

The performance of the proposed method was demonstrated by a flexible jobshop scheduling problem. Two approaches were applied to solve the integrated problem: the continuous-time scheduling model with linear piecewise Pareto curves and the discrete-time scheduling model with discretized Pareto points. The two approaches had similar performance in terms of the computational time and the optimal function value. The proposed method produced a profit 23.0% larger than the sequential method because the proposed integrated method could optimize the recipe and the scheduling decisions simultaneously. Comparing with the simultaneous integration method, the proposed method was much more efficient. The computational time was shorter by three orders of magnitudes while the objective function value was only 3.3% less than those of the simultaneous method. The efficiency ensured the online implementation of the proposed method under uncertainties. The proposed method was applied to a simulated example with uncertainty of equipment breakdown. Under the uncertainty, both scheduling decisions and recipe were able to be reoptimized online.

Future work will extend the proposed method to solve a batch scheduling problem with a general network structure. Because the scheduling problem itself can be very difficult to solve online for a batch process with a general network structure, we need to reduce the complexity not only for the integrated problem but also for the scheduling problem.

Notation

Operation

$\lvert\cdot\rvert$ =	size of set

Index

I =	order
j =	processing unit
k =	time slot in continuous-time scheduling model
l =	processing stage
m =	product
n =	finite element
p, q =	collocation point
r =	discrete points on Pareto frontier
t =	time point in discrete-time scheduling model

Set

I =	orders
I_P =	products, a subset of I
IL_j =	order stages, indexed by (i,l), which can be processed by unit j
J_{il} =	units capable for processing order i at stage l
K_j =	time slots for unit j
L_i =	operaional stages for order i
R_i =	discrete Pareto points for order i

Parameter

a_{pq} =	collocation matrix
b_p =	collocation vector
c_p =	collocation vector
c^{Fix} =	total fixed cost
c_i^{Fix} =	fixed cost for order i
c_i^P =	price for order i
d_i^I =	first due date function for order i
d_i^{II} =	second due date for order i
d_i^{ε} =	step size of discretizing Pareto frontier for order i
d_t =	step size of discrete-time scheduling model
n_f =	number of finite elements
n_i^{ε} =	number of discrete points in Pareto frontier
n_p =	number of collocation points
$s(i)$ =	initial value of material for order i
st_{il} =	setup time of stage l for order i
t_H =	upper bound of scheduling horizon
x_i^0 =	initial condition for order i

Greek letter

ε_i =	ε-value of multiobjective optimization for order i
$\varepsilon_i^{(r)}$ =	discrete value of ε_i
$\eta_i(\cdot)$ =	optimal-value function of dynamic optimization for order i
φ_{il} =	cost in dynamic model of stage l for order i
λ_{ir} =	SOS2 variables in piecewise linear Pareto curve
$\mu_i(\cdot)$ =	recipe function of total processing cost for order i
$\tilde{\mu}_i(\cdot)$ =	approximated function of $\mu_i(\cdot)$
$\pi_{il}(\cdot)$ =	recipe function of processing time in stage l for order i
$\tilde{\pi}_{il}(\cdot)$ =	approximated function of $\pi_{il}(\cdot)$
σ_{il} =	length of finite element of stage l for order i
τ_{il} =	ending time in dynamic model of stage l for order i
θ_{ijl} =	parameters of dynamic system at stage l of order i in unit j

Binary variable

β_{ir} =	representing SOS2 variables in piecewise linear Pareto curve
OW_i^I =	indicator of first segment in order price function for order i
OW_i^{II} =	indicator of second segment in order price function for order i
W_{ijkl} =	assignment of operational stage l of order i to time slot k of unit j (continuous-time scheduling model)
W_{ijlrt}^D =	assignment of operational stage l of order i to unit j starting from time t and using processing time indexed by r (discrete-time scheduling model)

Variable

C_i^{Var} =	variable cost for order i
$Cost$ =	total production cost
DP_{ir}^C =	total processing cost order i with mode r in discrete-time scheduling problem
DPT_{ilr} =	processing time of stage l for order i with mode r in discrete-time scheduling problem
DT_i^I =	first component of delivery date for order i
DT_i^I =	second component of delivery date for order i
DT_i^{III} =	third component of delivery date for order i
$k_{il}^{(p)}$, $k_{il}^{(q)}$ =	collocation point
ODT_i =	order delivery date of order i
P_i^C =	total processing cost for order i
$P_i^{C(r)}$ =	discrete value of P_i^C
PC_{il} =	processing cost of stage l for order i
$Profit$ =	production profit
PT_{il} =	processing time of stage l for order i
$PT_{il}^{(r)}$ =	discrete value of PT_{il}
S_{jk} =	slack variables for slot k in unit j
$Sales$ =	sales of all orders
t_{il} =	time variable in dynamic model of stage l for order i
$t_{il}^{(n)}$ =	discretized time variable in dynamic model of stage l for order i
TEI_{il} =	ending time of stage l for order i
TEJ_{jk} =	ending time of slot k in unit j
TSI_{il} =	starting time of stage l for order i
TSJ_{jk} =	starting time of slot k in unit j
u_{il} =	inputs in dynamic model of stage l for order i
$u_{il}^{(n)}$ =	discretized inputs in dynamic model of stage l for order i
UT_{ijkl} =	term for linearizing $WT_{ijkl}PT_{il}$
V^{Sch} =	collection of variables in scheduling problem
V_i^{Dyn} =	collection of variables in dynamic model for order i
WT_{ijkl} =	term for linearizing $WT_{ijkl}PT_{il}$
x_{il} =	state variables in dynamic model of stage l for order i
$x_{il}^{(n)}$ =	discretized state variables in dynamic model of stage l for order i
x_i^f =	final condition for order i
y_{il} =	output variables in dynamic model of stage l for order i
$y_{il}^{(n)}$ =	discretized output variables in dynamic model of stage l for order i

Acknowledgment

The authors thank Dr. John M. Wassick (The Dow Chemical Company, USA) for helpful discussions and comments.

Literature Cited

1. Grossmann I. Enterprise-wide optimization: a new frontier in process systems engineering. *AICHE J*. 2005;51(7):1846–1857.
2. Varma VA, Reklaitis GV, Blau GE, Pekny JF. Enterprise-wide modeling & optimization—an overview of emerging research challenges and opportunities. *Comput Chem Eng*. 2007;31(5-6):692–711.
3. Wassick JM. Enterprise-wide optimization in an integrated chemical complex. *Comput Chem Eng*. 2009;33(12):1950–1963.
4. Wassick JM, Agarwal A, Akiya N, Ferrio J, Bury S, You F. Addressing the operational challenges in the development, manufacture, and supply of advanced materials and performance products. *Comput Chem Eng*. 2012;47:157–169.
5. Maravelias CT, Sung C. Integration of production planning and scheduling: overview, challenges and opportunities. *Comput Chem Eng*. 2009;33(12):1919–1930.
6. Verderame PM, Elia JA, Li J, Floudas CA. Planning and scheduling under uncertainty: a review across multiple sectors. *Ind Eng Chem Res*. 2010;49(9):3993–4017.
7. Karimi IA, McDonald CM. Planning and scheduling of parallel semicontinuous processes .2. Short-term scheduling. *Ind Eng Chem Res*. 1997;36(7):2701–2714.
8. Sahinidis NV, Grossmann IE. Reformulation of multiperiod MILP models for planning and scheduling of chemical processes. *Comput Chem Eng*. 1991;15(4):255–272.
9. Pinto JM, Joly M, Moro LFL. Planning and scheduling models for refinery operations. *Comput Chem Eng*. 2000;24(9-10):2259–2276.
10. Yue D, You F. Planning and scheduling of flexible process networks under uncertainty with stochastic inventory: MINLP models and algorithm. *AIChE J*. In press. DOI: 10.1002/aic.13924
11. You FQ, Grossmann IE. Design of responsive supply chains under demand uncertainty. *Comput Chem Eng*. 2008;32(12):3090–3111.
12. Castro PM, Barbosa-Povoa AP, Novais AQ. Simultaneous design and scheduling of multipurpose plants using resource task network based continuous-time formulations. *Ind Eng Chem Res*. 2005;44(2):343–357.
13. Stefanis SK, Livingston AG, Pistikopoulos EN. Environmental impact considerations in the optimal design and scheduling of batch processes. *Comput Chem Eng*. 1997;21(10):1073–1094.
14. Terrazas-Moreno S, Flores-Tlacuahuac A, Grossmann IE. Simultaneous design, scheduling, and optimal control of a methyl-methacrylate continuous polymerization reactor. *AIChE J*. 2008;54(12):3160–3170.
15. Sakizlis V, Perkins JD, Pistikopoulos EN. Recent advances in optimization-based simultaneous process and control design. *Comput Chem Eng*. 2004;28(10):2069–2086.
16. Luyben ML, Floudas CA. Analyzing the interaction of design and control—1. *Amultiobjectiveframework and application to binarydistillationsynthesis*. *Comput Chem Eng*. 1994;18(10):933–969.
17. Nikacevic NM, Huesman AEM, Van den Hof PMJ, Stankiewicz AI. Opportunities and challenges for process control in process intensification. *Chem Eng Process*. 2012;52:1–15.
18. Perkins JD, Walsh SPK. Optimization as a tool for design/control integration. *Comput Chem Eng*. 1996;20(4):315–323.
19. Ricardez-Sandoval LA, Budman HM, Douglas PL. Integration of design and control for chemical processes: a review of the literature and some recent results. *Annu Rev Control*. 2009;33(2):158–171.
20. Harjunkoski I, Nystrom R, Horch A. Integration of scheduling and control-Theory or practice? *Comput Chem Eng*. 2009;33(12):1909–1918.
21. Munoz E, Capon-Garcia E, Moreno-Benito M, Espuna A, Puigjaner L. Scheduling and control decision-making under an integrated information environment. *Comput Chem Eng*. 2011;35(5):774–786.
22. Flores-Tlacuahuac A, Grossmann IE. Simultaneous cyclic scheduling and control of a multiproduct CSTR. *Ind Eng Chem Res*. 2006;45(20):6698–6712.
23. Nystrom RH, Franke R, Harjunkoski I, Kroll A. Production campaign planning including grade transition sequencing and dynamic optimization. *Comput Chem Eng*. 2005;29(10):2163–2179.
24. Busch J, Oldenburg J, Santos M, Cruse A, Marquardt W. Dynamic predictive scheduling of operational strategies for continuous processes using mixed-logic dynamic optimization. *Comput Chem Eng*. 2007;31(5-6):574–587.
25. Zhuge JJ, Ierapetritou MG. Integration of scheduling and control with closed loop implementation. *Ind Eng Chem Res*. 2012;51(25):8550–8565.
26. Chu Y, You F. Integration of scheduling and control with online closed-loop implementation: fast computational strategy and large-scale global optimization algorithm. *Comput Chem Eng*. 2012;47:248–268.
27. Shobrys DE, White DC. Planning, scheduling and control systems: why cannot they work together. *Comput Chem Eng*. 2002;26(2):149–160.
28. Allgor RJ, Barton PI. Mixed-integer dynamic optimization I: problem formulation. *Comput Chem Eng*. 1999;23(4-5):567–584.
29. Bansal V, Sakizlis V, Ross R, Perkins JD, Pistikopoulos EN. New algorithms for mixed-integer dynamic optimization. *Comput Chem Eng*. 2003;27(5):647–668.
30. Cuthrell JE, Biegler LT. Simultaneous optimization and solution methods for batch reactor control profiles. *Comput Chem Eng*. 1989;13(1-2):49–62.
31. Biegler LT. An overview of simultaneous strategies for dynamic optimization. *Chem Eng Process*. 2007;46(11):1043–1053.
32. Capon-Garcia E, Moreno-Benito M, Espuna A. Improved short-term batch scheduling flexibility using variable recipes. *Ind Eng Chem Res*. 2011;50(9):4983–4992.
33. Nie YS, Biegler LT, Wassick JM. Integrated scheduling and dynamic optimization of batch processes using state equipment networks. *AIChE J*. 2012;58(11):3416–3432.
34. Mishra BV, Mayer E, Raisch J, Kienle A. Short-term scheduling of batch processes. A comparative study of different approaches. *Ind Eng Chem Res*. 2005;44(11):4022–4034.
35. Pinto JM, Grossmann IE. A continuous time mixed integer linear programming model for short term scheduling of multistage batch plants. *Ind Eng Chem Res*. 1995;34(9):3037–3051.
36. Wassick JM, Ferrio J. Extending the resource task network for industrial applications. *Comput Chem Eng*. 2011;35(10):2124–2140.
37. Mendez CA, Cerda J, Grossmann IE, Harjunkoski I, Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput Chem Eng*. 2006;30(6-7):913–946.
38. Floudas CA, Lin XX. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Comput Chem Eng*. 2004;28(11):2109–2129.
39. Sundaramoorthy A, Maravelias CT. Computational study of network-based mixed-integer programming approaches for chemical production scheduling. *Ind Eng Chem. Res*. 2011;50(9):5023–5040.
40. Castro PM, Barbosa-Povoa AP, Matos HA. Optimal periodic scheduling of batch plants using RTN-based discrete and continuous-time formulations: a case study approach. *Ind Eng Chem Res*. 2003;42(14):3346–3360.
41. Vin JP, Ierapetritou MG. A new approach for efficient rescheduling of multiproduct batch plants. *Ind Eng Chem Res*. 2000;39(11):4228–4238.
42. Marchetti PA, Mendez CA, Cerda J. Simultaneous lot sizing and scheduling of multistage batch processes handling multiple orders per product. *Ind Eng Chem Res*. 2012;51(16):5762–5780.
43. Pinedo ML. Scheduling: Theory, Algorithms, and Systems. New York: Springer; 2012.
44. Pantelides CC. Unified frameworks for optimal process planning and scheduling. Foundations of Computer-Aided Process Operations. New York: Cache Publications; 1994:253–274.
45. Biegler LT, Cervantes AM, Wachter A. Advances in simultaneous strategies for dynamic process optimization. *Chem Eng Sci*. 2002;57(4):575–593.
46. Zennaro M. Natural continuous extensions of Runge-Kutta methods. *Math Comput*. 1986;46(173):119–133.
47. Barton PI, Allgor RJ, Feehery WF, Galan S. Dynamic optimization in a discontinuous world. *Ind Eng Chem Res*. 1998;37(3):966–981.
48. You FQ, Leyffer S. Mixed-integer dynamic optimization for oil-spill response planning with integration of a dynamic oil weathering model. *AIChE J*. 2011;57(12):3555–3564.
49. Ierapetritou MG, Floudas CA. Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. *Ind Eng Chem Res*. 1998;37(11):4341–4359.
50. Croxton KL, Gendron B, Magnanti TL. A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems. *Manage. Sci*. 2003;49(9):1268–1273.
51. Kondili E, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch operations—I. MILP formulation. *Comput Chem Eng*. 1993;17(2):211–227.
52. Shah N, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch operations—II. Computational issues. *Comput Chem Eng*. 1993;17(2):229–244.
53. Flores-Tlacuahuac A, Grossmann IE. Simultaneous cyclic scheduling and control of tubular reactors: parallel production lines. *Ind Eng Chem Res*. 2011;50(13):8086–8096.

Appendix: Data for the Flexible Jobshop Scheduling Example

Table A1. Data of Scheduling Problem

Symbol	Description	Value	Unit
C_1^P	Price of product 1	3.5E3	\$/order
C_2^P	Price of product 2	3.0E3	\$/order
C_3^P	Price of product 3	1.0E3	\$/order
C_1^F	Fixed cost of product 1	1.0E3	\$/order
C_2^F	Fixed cost of product 2	1.0E3	\$/order
C_3^F	Fixed cost of product 3	0.5E3	\$/order
C^{HW}	Cost of heating water	20	\$/m^3
C^{CW}	Cost of heating water	2	\$/m^3
D_1^I	First due date value of product 1	6	h
D_1^{II}	Second due date value of product 1	9	h
D_2^I	First due date value of product 2	6	h
D_2^{II}	Second due date value of product 2	9	h
D_3^I	First due date value of product 3	6	h
D_3^{II}	Second due date value of product 3	9	h
O_1	Order demand of product 1	2	Order
O_2	Order demand of product 2	2	Order
O_3	Order demand of product 3	2	Order

Table A2. Fixed-Processing Time (h)

	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
Order 1, Order 2	1.0	–	0.7	–	1.0
Order 3, Order 4	–	0.9	–	1.5	
Order 5, Order 6	–	1.2	–	–	–

Table A3. Setup Time (h)

	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
Order 1, Order 2	0.1	0.1	0.1	0.1	0.1
Order 3, Order 4	0.1	0.1	0.1	0.1	–
Order 5, Order 6	0.1	0.1	–	–	–

Table A4. Models of Reaction Kinetics

Task	Description	Reaction Kinetics	Differential Equations
T_{12}	The second operational stage of the first product	$M_{11}+M_{12}\xrightarrow{k_{12}^a}I_1$ $M_{11}+I_1\xrightarrow{k_{12}^b}U_1$ $k_{12}^a=Z_{12}^a e^{-E_{12}^a/T_R}$ $k_{12}^b=Z_{12}^b e^{-E_{12}^b/T_R}$	$\frac{dC_{M_{11}}(t)}{dt}=-k_{12}^a C_{M_{11}}C_{M_{12}}-k_{12}^b C_{M_{11}}C_{I_1}$ $\frac{dC_{M_{12}}(t)}{dt}=-k_{12}^a C_{M_{11}}C_{M_{12}}$ $\frac{dC_{I_1}(t)}{dt}=k_{12}^a C_{M_{11}}C_{M_{12}}-k_{12}^b C_{M_{11}}C_{I_1}$ $\frac{dC_{U_1}(t)}{dt}=k_{12}^b C_{M_{11}}C_{I_1}$
T_{14}	The fourth operational stage of the first product	$I_1\xrightarrow{k_{14}}P_1$ $k_{14}=Z_{14}e^{-E_{14}/T_R}$	$\frac{dC_{I_1}(t)}{dt}=-k_{14}C_{I_1}$ $\frac{dC_{P_1}(t)}{dt}=k_{14}C_{I_1}$
T_{21}	The first operational stage of the second product	$M_2\underset{k_{21}^b}{\overset{k_{21}^a}{\rightleftarrows}}I_2$ $I_2\xrightarrow{k_{21}^c}U_2$ $k_{21}^a=Z_{21}^a e^{-E_{21}^a/T_R}$ $k_{21}^b=Z_{21}^b e^{-E_{21}^b/T_R}$ $k_{21}^c=Z_{21}^c e^{-E_{21}^c/T_R}$	$\frac{dC_{M_2}(t)}{dt}=-k_{21}^a C_{M_2}+k_{21}^b C_{I_2}$ $\frac{dC_{I_2}(t)}{dt}=k_{21}^a C_{M_2}-\left(k_{21}^b+k_{21}^c\right)C_{I_2}$ $\frac{dC_{U_2}(t)}{dt}=k_{21}^c C_{I_2}$
T_{23}	The third operational stage of the second product	$I_2\xrightarrow{k_{22}}P_2 k_{23}=Z_{23}e^{-E_{23}/T_R}$	$\frac{dC_{I_2}(t)}{dt}=-k_{23}C_{I_2}$ $\frac{dC_{P_2}(t)}{dt}=k_{23}C_{I_2}$
T_{31}	The first operational stage of the third product	$M_3\xrightarrow{k_{31}}P_3 k_{31}=Z_{31}e^{-E_{31}/T_R}$	$\frac{dC_{M_3}(t)}{dt}=-k_{31}C_{M_3}$ $\frac{dC_{P_3}(t)}{dt}=k_{31}C_{M_3}$

Table A5. Models of the Heat Transfer between the Reactor and the Jacket

Temperature	Differential Equation
Reactor	$\frac{d}{dt}T_R(t)=\frac{\sum_{i=1}^{n} r_i(-\Delta H_i)}{\rho_R c_R}+\frac{UA(T_J-T_R)}{v_R\rho_R c_R}$
Jacket	$\frac{d}{dt}T_J(t)=\frac{F_H T_H+F_C T_C-(F_H+F_C)T_J}{V_J}+\frac{UA(T_R-T_J)}{V_J\rho_J c_J}$

Table A6. Parameters of Reaction Tasks

Task	Symbol	Description	Value	Unit
T_{12}	Z_{12}^a	Frequency factor	4E3	$m^3/(mol\ h)$
	Z_{12}^b	Frequency factor	1E8	$m^3/(mol\ h)$
	E_{12}^a	Normalized activation energy	5E3	K
	E_{12}^b	Normalized activation energy	1E4	K
	ΔH_{12}^a	Heat of reaction	−30	kJ/mol
	ΔH_{12}^b	Heat of reaction	−20	kJ/mol
	ρ_R^{12}	Density of reactor	8E2	kg/m^3
	$C_{P_R}^{12}$	Heat capacity of reactor	3	kJ/(kg K)
T_{13}	Z_{13}	Frequency factor	1E4	$m^3/(mol\ h)$
	E_{13}	Normalized activation energy	3E3	K
	ΔH_{13}	Heat of reaction	0	kJ/mol
	ρ_R^{13}	Density of reactor	8E2	kg/m^3
	$C_{P_R}^{13}$	Heat capacity of reactor	3	kJ/(kg K)
T_{21}	Z_{21}^a	Frequency factor	1E3	$m^3/(mol\ h)$
	Z_{21}^b	Frequency factor	5E2	$m^3/(mol\ h)$
	Z_{21}^c	Frequency factor	2E4	$m^3/(mol\ h)$
	E_{21}^a	Normalized activation energy	2E3	K
	E_{21}^b	Normalized activation energy	3E3	K
	E_{21}^c	Normalized activation energy	4E3	K
	ΔH_{21}^a	Heat of reaction	−20	kJ/mol
	ΔH_{21}^b	Heat of reaction	0	kJ/mol
	ΔH_{21}^c	Heat of reaction	−10	kJ/mol
	ρ_R^{21}	Density of reactor	1.2E3	kg/m^3
	$C_{P_R}^{21}$	Heat capacity of reactor	3.5	kJ/(kg K)
T_{22}	Z_{22}	Frequency factor	5E2	$m^3/(mol\ h)$
	E_{22}	Normalized activation energy	2E3	K
	ΔH_{22}	Heat of reaction	0	kJ/mol
	ρ_R^{22}	Density of reactor	1.2E3	kg/m^3
	$C_{P_R}^{22}$	Heat capacity of reactor	3.5	kJ/(kg K)
T_{31}	Z_{31}	Frequency factor	1E2	$m^3/(mol\ h)$
	E_{31}	Normalized activation energy	1E3	K
	ΔH_{31}	Heat of reaction	10	kJ/mol
	ρ_R^{31}	Density of reactor	1E3	kg/m^3
	$C_{P_R}^{31}$	Heat capacity of reactor	4	kJ/(kg K)

The normalized activation energy is the activation energy divided by the gas constant.

Table A7. Parameters of Reactors

Reactor	Symbol	Description	Value	Unit
R_I	V_R^I	Volume of reactor	5	m^3
	V_J^I	Volume of jacket	1	m^3
	ρ_J^I	Density of jacket	1E3	kg/m^3
	$C_{P_J}^I$	Heat capacity of jacket	4.186	kJ/(kg K)
	UA^I	Heat-transfer coefficient	8E4	kJ/(h K)
	T_H^I	Temperature of heating water	370	K
	T_C^I	Temperature of cooling water	300	K
	$T_{R,\max}^I$	Maximum temperature of reactor	350	K
	$T_{J,\max}^I$	Maximum temperature of jacket	370	K
	$F_{\max}^I$	Maximum flow rate of heating and cooling water	20	m^3/h
R_{II}	V_R^{II}	Volume of reactor	5	m^3
	V_J^{II}	Volume of jacket	1.5	m^3
	ρ_J^{II}	Density of jacket	1E3	kg/m^3
	$C_{P_J}^{II}$	Heat capacity of jacket	4.186	kJ/(kg K)
	UA^{II}	Heat-transfer coefficient	1E5	kJ/(h K)
	T_H^{II}	Temperature of heating water	370	K
	T_C^{II}	Temperature of cooling water	300	K
	$T_{R,\max}^{II}$	Maximum temperature of reactor	360	K
	$T_{J,\max}^{I}$	Maximum temperature of jacket	370	K
	$F_{\max}^{II}$	Maximum flow rate of heating water	30	m^3/h

The parameter values of reactor R_{III} are identical to those of reactor R_{II}.

Table A8. Data of Materials and Products

Production Line	Initial Material Concentration	Required Product Concentration
1	$C_{M_{11}} = C_{M_{11}} = 5E3 \quad mol/m^3$	$C_{P_1} = 4E3 \quad mol/m^3$
2	$C_{M_2} = 2E3 \quad mol/m^3$	$C_{P_2} = 1.5E3 \quad mol/m^3$
3	$C_{M_3} = 1E3 \quad mol/m^3$	$C_{P_3} = 0.9E3 \quad mol/m^3$

Table A9. Temperature Specification

Production Line	Initial Temperature (K)	Final Temperature (K)
1	300	320
2	300	320
3	300	320

Manuscript received Sept. 14, 2012, and revision received Dec. 1, 2012.