Moving-horizon Method for Integrating Scheduling and Control

Yunfei Chu, Fengqi You *

Abstract—Online integration of scheduling and control is crucial to cope with process uncertainties. We propose a new online integrated method for sequential batch processes, where the integrated problem is solved to determine controller references rather than process inputs. To achieve the goal of computational efficiency and rescheduling stability, a rolling horizon approach is developed. A reduced integrated problem in a re-solving horizon is formulated, which can be solved efficiently online. Solving the reduced problem only changes a small part of the initial solution, guaranteeing rescheduling stability. The integrated method is demonstrated in a simulated case study.

I. INTRODUCTION

Enterprise-wide optimization (EWO) has become a major goal in the process industries [1, 2]. As two major decision-making layers in the production hierarchy, integration of scheduling and control has attracted significant research efforts in recent years [3-7]. Compared with the traditional method where the scheduling problem and the control problem are solved sequentially, the integrated method can optimize the overall performance of the production process.

However, most integrated methods focus on simultaneously optimizing the two problems offline. There are relatively few studies [8] which investigate how the integrated approach can be implemented online. There are three critical issues for an online integrated method, which are often neglected in existing methods.

The first issue is how the integrated method cooperates with unit controllers. In most existing methods, the controllers are completely incorporated into the integrated problem, which is solved to determine the process inputs directly. The integrated method solves a mixed-integer dynamic optimization (MIDO) [9]. The integration structure neglects process uncertainties. However, unforeseen disruptions can cause the actual production to deviate from what is planned [10].

Besides the integration structure, another issue which is often neglected is the computational efficiency. Most previous studies concentrate on the formulation of the integrated problem. The formulated MIDO problem is often discretized into a mixed-integer nonlinear programming (MINLP) problem which is then solved by a general-purpose MINLP solver [11]. This solution approach, called the simultaneous method, is straightforward. However, the batch scheduling problems are usually challenging to solve [12], and the addition of dynamic models in the MIDO problems makes the online, direct solution almost impossible.

The third important, while often neglected, issue for an online integrated method is the rescheduling stability. It is a measure of the discrepancy between the initial schedule and the rescheduling solution [13]. Different from the automatic control loops, execution of a schedule usually requires human resources. Frequent and dramatic change in the rescheduling solution can cause the problem of “floor nervousness”, substantially lowering the work efficiency. The existing integrated methods seldom take the rescheduling stability into account, which tend to change the entire schedule even if only a minor disruption occurs.

To address the three important online implementation issues, we propose a new online integrated method. To cooperate with the advanced controllers, a novel integration framework is developed in Fig. 1. The integrated problem is solved to determine the reference trajectories for controllers. The controllers then track the references by manipulating the process inputs in the real-time feedback loops. The derivation between the real measured values and the predetermined values reflects the uncertainties. At the higher level, the integrated problem is solved online to determine new reference trajectories along with scheduling decisions.

Besides the integration structure, we develop a rolling horizon approach to solve the integrated problem online, which addresses computational efficiency and rescheduling stability at the same time. Under uncertainties, this policy determines a new solution of the integrated problem based on the initial one (the offline solution or the previous online solution). Only a reduced problem in a short horizon is solved while the scheduling decisions and controller references beyond the horizon follow those of the initial solution. Dramatic change in the rescheduling solution is avoided. Another benefit from the rolling horizon policy is that only a reduced problem is solved which is much simpler than the entire problem.

It should be noted that we use the rolling horizon approach to solve the integrated problem online, which is distinct from those for solving scheduling problems [14-16]. Beside the scheduling model, the integrated problem

Fig. 1. Structures for integrated scheduling and control problem

Y. Chu is with the Department of Chemical & Biological Engineering, Northwestern University, Evanston, IL 60208 USA.
F. You is with the Department of Chemical & Biological Engineering, Northwestern University, Evanston, IL 60208 USA (phone: 847-467-2943; fax: 847-491-3728; e-mail: you@northwestern.edu).
includes a number of dynamic models. The online integrated problem determines not only scheduling decisions but also controller references.

II. PROBLEM STATEMENT

This work concentrates on the sequential batch process where material splitting and mixing is prohibited [17] and the scheduling problem has relatively simple structure [18].

---

**Implementation:**
To solve the integrated scheduling and control problem online under uncertainty

**Process structure:**
Sequential batch process

**Controllers:**
Stable controllers with good tracking performance

**Given:**
Rescheduling horizon
Job release date and due date
Tasks and capable processing units
Sale price and fixed cost of completing a job
Unit cost of utilities for executing a task
Dynamic models for a task processed in a unit

**Determine:**
Task assignment and sequence
Task processing times, and processing costs
Job completion times
Reference trajectories for closed-loop control systems

**Objective:**
To maximize the production profit

---

III. ONLINE INTEGRATION FRAMEWORK

To achieve the goal of both computational efficiency and rescheduling stability, we propose a rolling horizon approach. It is a solution repairing approach under uncertainties. The rolling horizon approach is based on an initial solution. Instead of changing the entire solution completely, the rolling horizon approach only modifies a small portion of the initial solution.

The horizon, in which the integrated problem is resolved, stretches from the current time point at which the information about the disruptions is received. The horizon length is equal to a predetermined value. To emphasize the difference between the integrated problem and a scheduling problem, we name the horizon as the “re-solving horizon” instead of the common rescheduling horizon.

The flow chart of the rolling horizon approach is shown in Fig. 2. Initially, the integrated scheduling and dynamic optimization problem is solved offline. The solution includes the scheduling decisions and the controller references. Then the production is carried out according to the offline schedule and the process dynamic systems are manipulated by the controllers tracking the reference trajectories.

When disruptions occur in the process, the controllers send event messages to trigger the solution of the integrated problem online. Each event message records the information about the disruptions, e.g. the changed processing times. In this work, we focus on non-preemptive tasks. The controller of a task only sends the message after the task is completed or when the task is forcefully terminated by unit breakdown.

When the integrated method receives an event message, it updates the variables in the integrated problem according to the actual data. The disruptions can cause infeasibility of the initial schedule. For example, the prolonged processing time of a task makes it impossible to start the subsequent task at the scheduled starting time. The infeasibility is handled by shifting the task bars on the Gantt to the right. The shifted distance is kept as small as possible. The shifted feasible schedule serves as a good starting point from which the reduced problem is formulated and solved.

Based on the shifted schedule, the reduced problem in the rescheduling horizon is formulated. The detailed formulation is presented in the next section. The assignment and sequence of the tasks inside the rescheduling horizon can be changed as well as the recipe data.

---

IV. FORMULATION OF ONLINE INTEGRATED PROBLEM

The scheduling problem is formulated by the general precedence model [19]. Let a job be indexed by \( j \) and a stage by \( k \). A task, representing a job operational stage, is denoted by the pair \((j,k)\). A processing unit is indexed by \( i \) and the units capable of executing task \((j,k)\) belongs to set \( I_{jk} \).

The scheduling model includes two types of binary variables: the assignment variables \( \xi_{ik} \), which indicate if task \((j,k)\) is assigned to unit \( i \), and the precedence variables \( \beta_{ok} \), which are equal to one if task \((j,k)\) precedes task \((j',k')\) in unit \( i \). The continuous variables in the scheduling model include the
task starting times $T_{S\mu}$, the task ending times $T_{E\mu}$. For the tasks with fixed recipe, the parameter processing times are denoted by $p_{t\mu}$, while for the tasks having variable recipe, the variable processing times are denoted by $PT_{\mu}$.

The online integrated problem is a reduced one derived from the initial solution. According to the re-solving horizon, the tasks in the initial solution can be clustered into three groups as

$$ JK = \{ \text{Historical tasks}\} $$

$$ JK = \{ \text{Re-solving tasks}\} $$

$$ JK = \{ \text{Future tasks}\} $$

We use a variable with two bars above to represent the historical data of the variable and use a variable with one bar above to represent its value in the initial solution. For example, let $T_{S\mu}$ denote the starting time of task $(j, k)$. Then $\overline{T_{S\mu}}$ denotes the historical data of the starting time and $\overline{T_{S\mu}}$ denotes the value in the initial solution.

**Model in historical period**

In the historical period, all variables are fixed according to the data from tasks that are already completed or still being executed. These tasks belong to set $JK$. As the assignment of the historical tasks is known, the binary assignment variables are fixed according to the historical data

$$ \xi_{\mu} = \overline{\xi_{\mu}}, \forall (j, k) \in JK, i \in I_\mu $$

Similarly, the binary precedence variables $\beta_{ij/k}$ are fixed at

$$ \beta_{ij/k} = \overline{\beta_{ij/k}}, \forall (j, k) \in JK, (j', k') \in JK, i \in I_\mu \cap I_{j'} $$

As the tasks in the historical period have all been started, the task starting times are fixed according to the historical data

$$ T_{S\mu} = \overline{T_{S\mu}}, \forall (j, k) \in JK $$

When the tasks are completed, the ending times are fixed according to the historical data

$$ T_{E\mu} = \overline{T_{E\mu}}, \forall (j, k) \in JK, (j, k) \text{ is completed} $$

When the tasks are still being executed, the ending times are fixed at

$$ T_{E\mu} = \overline{T_{S\mu}} + \sum_{i \in I_\mu} \overline{\xi_{\mu}} PT_{\mu} $$

$$ \forall (j, k) \in JK, (j, k) \text{ is not completed} $$

where $\overline{PT_{\mu}}$ is the processing time of the initial solution, representing the expected processing time.

**Model in re-solving horizon**

In the re-solving horizon, the integrated problem is solved for the tasks belonging to set $JK$. The integrated problem is a reduced formulation of the original entire problem. The tasks in the re-solving horizon can be rescheduled and the operational recipe can be re-optimized. To distinguish the tasks with dynamic models from the ones without, the task set is partitioned into two subsets as

$$ JK = JK^D \cup JK^F $$

where

$$ JK^D = \{(j, k) \in JK \mid \text{Having dynamic model}\} $$

$$ JK^F = \{(j, k) \in JK \mid \text{Having fixed recipe}\} $$

A task in $JK^D$ has the variable processing time $PT_{\mu}$ and the variable processing cost $PC_{\mu}$, which can be manipulated by changing the controller reference. By contrast, a task belonging to $JK^F$ has the fixed processing time, which is a known parameter denoted by $p_{t\mu}$.

For a task with the fixed recipe, the task ending time $T_{E\mu}$ is equal to the starting time $T_{S\mu}$ plus the fixed processing time $p_{t\mu}$,

$$ T_{E\mu} = T_{S\mu} + \sum_{i \in I_\mu} \xi_{\mu} p_{t\mu}, \forall (j, k) \in JK^F $$

Similarly, the ending time for a task with variable recipe is

$$ T_{E\mu} = T_{S\mu} + \sum_{i \in I_\mu} \xi_{\mu} PT_{\mu}, \forall (j, k) \in JK^D $$

where $PT_{\mu}$ is the processing time. By linearizing the product $XPT_{\mu} = \xi_{\mu} PT_{\mu}$, the nonlinear constraints can be formulated into linear ones as shown below

$$ T_{E\mu} = T_{S\mu} + \sum_{i \in I_\mu} XPT_{\mu}, \forall (j, k) \in JK^D $$

$$ 0 \leq XPT_{\mu} \leq PT_{\mu}, \forall (j, k) \in JK^D, i \in I_\mu $$

$$ XPT_{\mu} \leq \xi_{\mu} p_{t\mu}, \forall (j, k) \in JK^D, i \in I_\mu $$

$$ XPT_{\mu} \geq PT_{\mu} - (1 - \xi_{\mu}) p_{t\mu}, \forall (j, k) \in JK^D, i \in I_\mu $$

where $p_{t\mu}$ is an upper bound of $PT_{\mu}$. Because the tasks for a job are executed sequentially through the operational stages, the task in the current stage cannot start earlier than the ending time of the task in the previous stage,

$$ T_{S\mu} \geq T_{E_{(j,k)}}, \forall (j, k) \in JK, k \geq 2 $$

The assignment and the sequence of the tasks in the re-solving horizon can be changed. The assignment variables $\xi_{\mu}$ are constrained by

$$ \sum_{i \in I_\mu} \xi_{\mu} = 1, \forall (j, k) \in JK $$

because each task is executed exactly once. In a processing unit, a task is executed either before or after another one, so the starting times of two tasks are constrained by the precedence variables $\beta_{ij/k}$

$$ T_{S\mu} + p_{t\mu} - b_{\mu}(1 - \beta_{ij/k}) \leq T_{S_{(j',k')}} $$

$$ \forall (j, k) \in JK^F, (j', k') \in JK^F, (j, k) \neq (j', k') $$

2950
\[ TS_{ik} + PT_{ik} - b_m (1 - \beta_{ijk}) \leq TS_{j'k'} \]
\[ \forall (j,k) \in JK', \ (j',k') \in JK', \ (j,k) \neq (j',k') \] (14)

where \( b_m \) denotes a big-M term. The precedence variables are constrained by the assignment variables

\[ \xi_{ik} + \xi_{ik'} - \beta_{ijk} - \beta_{ijk'} \leq 1, \ \forall (j,k) \in JK \]
\[ \forall (j,k) \in JK', \ (j',k') \in JK', \ (j,k) \neq (j',k') \] (15)

Only the tasks belonging to \( JK^D \) have dynamic models. The dynamic model of a task is described by a set of differential equations

\[ \frac{dX_{ik}(T_{ik})}{dT_{ik}} = F_{ik}(X_{ik}(T_{ik}), U_{ik}(T_{ik})) \] (16)

The differential equation (16) is indexed by unit \( i \), and task \( (j,k) \). Concerningly, all variables in the differential equations are indexed by \( ijk \), including the time variable \( T_{ik} \). For a compact expression, the states \( X_{ijk}(T_{ik}) \) and the inputs \( U_{ijk}(T_{ik}) \) are expressed by the vector forms. The vector forms are also used to express other variables and constraints in the dynamic models.

Using the collocation method [20], the continuous-time state and input trajectories satisfying the differential equations are discretized. The discretization procedure transforms the differential equation (16) into algebraic equations as

\[ X_{ik}^r = X_{ik}^{r+1} + L_{ijk} \sum_q a_{eq} F_{ik}(X_{ik}^{r+1}, U_{ik}^r) \] (17)
\[ X_{ik}^r = X_{ik}^{r+1} + L_{ijk} \sum_q b_{eq} F_{ik}(X_{ik}^{r+1}, U_{ik}^r) \] (18)

where \( r \) denotes finite elements, \( q \) and \( q' \) denote collocation points. \( X_{ik}^r \) and \( X_{ik}^{r+1} \) denote the discretized state value at a collocation point and \( U_{ik}^r \) is the discretized value of the input trajectory. The parameter \( a_{eq} \) represents a coefficient in the collocation matrix and \( b_{eq} \) is an element in the collocation vector. The discretization procedure is actually a numerical integration method for solving the differential equations.

Optimizing the dynamic models returns processing times and processing costs which are recipe data for the scheduling model. Without loss of generality, the processing cost can be represented as a function \( \phi_{ijk} \) of the final state value

\[ PC_{ijk} = \phi_{ijk}(X_{ijk}^{r+1}) \] (19)

The processing time is equal to the product

\[ PT_{ik} = L_{ijk} r_{ijk}^r \] (20)

**Model in future period beyond re-solving horizon**

For the future tasks beyond the re-solving horizon, the binary variables and the processing times are fixed at those determined by the initial solution. The starting times can be varied to guarantee the feasibility while the processing times are fixed according to the initial solution.

The tasks in this period belong to \( JK' \). The assignment variables are fixed at

\[ \xi_{ik} = \bar{\xi}_{ik}, \ \forall (j,k) \in JK', i \in I_{jk} \] (21)

where \( \bar{\xi}_{ik} \) is the value of the binary variable in the initial solution. The precedence variables are fixed at the initial solution value \( \bar{\beta}_{ijk} \).

\[ \beta_{ijk} = \bar{\beta}_{ijk}, \ \forall (j,k) \in JK', \ (j',k') \in JK' \] (22)

The task processing times are fixed at

\[ PT_{ik} = \bar{PT}_{ik}, \ \forall (j,k) \in JK', i \in I_{jk} \] (23)

The task starting times \( TS_{ik} \) and the ending times \( TE_{jk} \) are variables, which are constrained by

\[ TE_{jk} = TS_{ij} + \sum_{i \in i_{jk}} \xi_{ik} PT_{ik}, \ \forall (j,k) \in JK' \] (24)
\[ TE_{jk} = TS_{ij} + \sum_{i \in i_{jk}} \xi_{ik} PT_{ik}, \ \forall (j,k) \in JK' \] (25)

where set \( JK' \) indicates the tasks with the fixed recipe and set \( JK'' \) includes those with the variable recipe. The two sets partition \( JK \), i.e. \( JK' \cup JK'' = JK \).

**Objective function**

The objective function for the integrated problem is to maximize the profit [21], which is the difference between the sales and the costs

\[ \max Profit = Sales - Cost' - c^r. \] (26)

The sales is equal to the sum of the job prices

\[ Sales = \sum_j PR_j. \] (27)

The variable cost is the sum of processing costs for the tasks with variable recipe

\[ Cost' = \sum_{(j,k) \in JK', i \in i_{jk}} \xi_{ik} PC_{ijk}. \] (28)

The bilinear terms can be linearized by introducing continuous variables \( XPC_{ijk} = \xi_{ik} PC_{ijk} \) such that

\[ 0 \leq XPC_{ijk} \leq PC_{ijk} \] (29)
\[ XPC_{ijk} \leq \xi_{ik} PC_{ijk}^{max} \] (30)
\[ XPC_{ijk} \geq PC_{ijk} - (1 - \xi_{ik}) PC_{ijk}^{max} \] (31)

The job prices \( PR_i \) are functions on the job completion times \( DT_{ij} \). There is a threshold \( d_j \) on each completion time. When the completion time is less than the threshold value, the job has a constant price \( pr_j \). However, when the completion time exceeds the threshold, a linear penalty term
is added and the job price reduces gradually as the completion time increases. The completion time cannot exceed the upper bound $c_j^{\text{max}}$.

The penalty on the completion time results in a piecewise linear job price function, which has two segments. Binary variables $\gamma_j$ are introduced to indicate if a job completion time exceeds the threshold ($\gamma_j = 1$) or not ($\gamma_j = 0$). Then the job completion times can be expressed by

$$DT_j = DT_j^0 + DT_j^u, \ \forall j$$

$$\gamma_j d_j \leq DT_j^0 \leq d_j, \ \forall j$$

$$0 \leq DT_j^u \leq \gamma_j (c_j^{\text{max}} - d_j), \ \forall j$$

The job completion time is equal to the ending time of the last task

$$DT_j = TE_j + DT_j^v, \ \forall j$$

The completion times are constrained by the upper bound

$$DT_j \leq c_j^{\text{max}}, \ \forall j$$

The job price is equal to

$$PR_j = pr_j \left(1 - \frac{1}{c_j^{\text{max}} - d_j}DT_j^u\right), \ \forall j$$

V. CASE STUDY

To demonstrate the proposed online integrated method, we apply it to a simulated example. The process diagram is shown in Fig. 3. The batch process consists of three operational stages: a reaction task, a filtration task, and another reaction task. The first reaction task can be executed in Reactor R1 or RII and the second reaction task can be executed in Reactor RIII or RIV. The filtration task is processed in a filter F. The batch process aims to complete 8 jobs. The filtration stage of a job has the fixed processing time and the processing cost. The two reaction stages of a job are described by dynamic models.

The total number of dynamic models in the integrated problem is $8 \times 2 \times 2 = 32$ (#jobs $\times$ #reaction tasks $\times$ #units). The differential equations of each dynamic model are discretized by the collocation method using 30 finite elements. The 32 dynamic models can generate a great number of nonlinear equations after the discretization procedure.

When uncertainties occur, the information concerning disruptions will be passed to the integrated problem. Then the integrated problem is re-solved online. In this case study, the time available for the integrated method is set as 0.1 hour (360 seconds). The time period is not enough to solve the entire integrated problem. The proposed method can, however, return a solution in a short time period because it merely solves a reduced problem.

To demonstrate the proposed method, we create an uncertainty scenario of the unit breakdown. In this scenario, the disruption of unit breakdown is investigated. Unit breakdown is a common uncertainty in scheduling problem. It cannot be cope with by a local control system and the integrated problem has to be solved online. In this example, the process is initially operated according to the offline solution. Reactor R3 breaks down from 0 to 1 hour, shown in Fig. 4. The breakdown delays the starting time of the first task (5,1) in the reactor. When the reactor is recovered at 1 hour, the integrated problem is solved by the proposed method.

First, the shifted rescheduling problem is solved. The shifted schedule is shown in Fig. 4 (a). The re-solving

---

![Fig. 3. Process diagram for the case study](image_url)

![Fig. 4. The scheduling results by resolving the integrated problem under uncertainty.](image_url)

The label on a task bar denotes (job, stage). Reactor RII breaks down from 0 to 1 hour. The integrated problem is solved when the reactor is recovered. The period between two dash lines in the shifted schedule is the re-solving horizon. The historical tasks are represented in gay and the future tasks beyond the re-solving horizon are represented by white boxes. The tasks in the re-solving horizon are colored.
The comparisons of the objective function values for the shifted schedule and the online solution are listed in Table 1. The reduced integrated problem includes 8942 equations 8426 continuous variables, and 138 binary variables. The online problem is reduced so that it can be solved to the 1% optimality gap in 151.3 seconds, which is less than the limit of 360 seconds. Under the uncertainty, the profit is degenerated from 429.4 m.u. of the offline schedule to 377.6 m.u. of the shifted schedule. The profit is degenerated by 12.1% for the shifted schedule while it is degenerated only by 1.5% for the online solution.

VI. CONCLUSION

We propose an online integrated method for the sequential batch process. The integrated method determines the controller references simultaneously with scheduling decisions and task recipes. To obtain an efficient solution while avoid dramatic changes from the initial solution, a rolling horizon approach is developed. A reduced online problem is formulated based on the initial solution. The reduced problem is much simpler than the entire integrated problem and can be solved efficiently. Since only a part of the initial solution is changed, rescheduling stability is guaranteed. The online integrated method is demonstrated by the case study including 8 jobs and 32 dynamic models. The online integrated problem in a re-solving horizon of 1 hour can be solved efficiently in 3 minutes. Under uncertainties of the control system disruption and the processing unit breakdown, the online solution prevents a large loss in the production profit. The decreased profit is only 1.5% for the integrated method while it is 12.1% for the simple shifted scheduling method.

REFERENCES