Computers and Chemical Engineering 72 (2015) 255–272

Contents lists available at ScienceDirect

Computers and Chemical Engineering

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

Integrated planning and scheduling under production uncertainties: Bi-level model formulation and hybrid solution method

Yunfei Chu[a], Fengqi You[a,*], John M. Wassick[b], Anshul Agarwal[b]

[a] *Department of Chemical and Biological Engineering, Northwestern University, Evanston, IL 60208, USA*
[b] *The Dow Chemical Company, Midland, MI 48674, USA*

ARTICLE INFO

Article history:
Received 24 November 2013
Received in revised form 24 January 2014
Accepted 19 February 2014
Available online 2 March 2014

Keywords:
Planning
Scheduling
Service-level constraints
Hybrid method
Agent-based modeling and simulation.

ABSTRACT

We propose a novel method for integrating planning and scheduling problems under production uncertainties. The integrated problem is formulated into a bi-level program. The planning problem is solved in the upper level, while the scheduling problems in the planning periods are solved under uncertainties in the lower level. The planning and scheduling problems are linked via service level constraints. To solve the integrated problem, a hybrid method is developed, which iterates between a mixed-integer linear programming solver for the planning problem and an agent-based reactive scheduling method. If the service level constraints are not met, a cutting plane constraint is generated by the agent-based scheduling method and appended to the planning problem which is solved to determine new production quantities. The hybrid method returns an optimality gap for validating the solution quality. The proposed method is demonstrated by two complicated problems which are solved efficiently with small gaps less than 1%.

1. Introduction

Planning and scheduling are two core decision layers in a manufacturing organization (Wassick et al., 2012). Owing to different objectives and time scales, planning and scheduling problems are often solved separately in a sequential way. For example, a planning problem can be solved to determine weekly production quantities according to the customer orders. The output of the planning problem, the weekly production quantities, are then passed to the scheduling problem and a production schedule in each week is determined, such as assigning an operational task to a capable unit and sequencing the tasks in every unit. The sequential nature of the traditional approach prevents the planning model from using detailed production schedule information as the scheduling problem is not solved in the planning phase. Instead, the planning problem is solved based on aggregate information which is commonly a rough approximation of the real production data. For example, a planning model often includes a parameter to denote the time required to process a product but the actual processing time, due to product transition policies, is highly dependent on the production schedule. Though widely applied for its simplicity, the sequential methods often result in a suboptimal solution for the entire production process, or even an infeasible production plan that cannot be fulfilled by the scheduling procedure (Shen, Wang, & Hao, 2006; Tan & Khoshnevis, 2000).

To overcome the drawbacks of sequential methods, a great variety of integrated methods have been developed (Birewar & Grossmann, 1990; Maravelias & Sung, 2009; Phanden, Jain, & Verma, 2011; Shao, Li, Gao, & Zhang, 2009), which aim to improve the overall performance of the entire planning process by collaboratively solving the production planning problem and the scheduling problem. A major category of the integrated methods is the simultaneous methods, which solve a monolithic model formulated by combining all constraints of the planning model and the scheduling model. The computational difficulty arising from the formulated complex model can be addressed by a decomposition method (Li & Ierapetritou, 2009). The simultaneous methods can theoretically obtain the global optimal solution for the entire process. However, they encounter some practical difficulties due to the computational challenges (Shobrys & White, 2002). Implementation of the simultaneous method may require dismantling and reorganizing the existing production hierarchy in a company (Pinedo, 2009).

Other types of integrated methods, based on bi-level programming, have been proposed in the literature (Ryu, Dua, & Pistikopoulos, 2004). The planning problem is the upper level problem while the scheduling problem is the lower level problem, which is consistent with the existing production hierarchy. A bi-level optimization problem is closely related with a Stackelberg game which is played between a leader and multiple followers (Colson, Marcotte, & Savard, 2007). The planning problem acts as the leader,

* Corresponding author. Tel.: +1 847 467 2943; fax: +1 847 491 3728.
E-mail address: you@northwestern.edu (F. You).

http://dx.doi.org/10.1016/j.compchemeng.2014.02.023
0098-1354/© 2014 Elsevier Ltd. All rights reserved.

Nomenclature

Index	
i	processing unit (unit agent)
j, j'	job (job agent)
m	sample in the Monte Carlo method
n	iteration in hybrid method
p	product
r	data set for the Monte Carlo method
s	element in vector
t	planning period
Parameter	
A_{pt}	unit processing time (aggregate information) for product p in period t
CAP_t	total processing time (aggregate information) of period t
C_p^H	unit hold cost for product p
C_p^S	unit setup cost for product p
D_{pt}	order demand of product p at the end of period t
E_s	s-th column of identity matrix
H_t	length of period t
M_{pt}	upper bound of w_{pt}
N_{MC}	number of sampling points in the Monte Carlo method
N_p	number of products
N_R	number of data sets
P_{MC}	probability with which the results by the Monte Carlo method satisfy the probability constraints
P_t	threshold of probabilistic constraint in period t in general formulation of integrated problem
PT_{ij}	nominal processing time of task for job j processed in unit i
S_t^L	threshold value of service level in period t
SUT_i	nominal startup time of unit i
$TT_{ij'j}$	nominal transition time from job j' to job j in unit i
Variable	
α_{tn}	coefficient vector of linear bounding function in period t for iteration n
β_{pt}	equal to 1 if product p is manufactured in period t
$cost$	total cost
ct_j	completion time of job agent JA_j
δ_{tn}	interception of linear bounding function in period t for iteration n
dt	time step in agent-based simulation
inv_{pt}	inventory level of product p at the end of period t
ms_t	makespan of scheduling problem in period t
pm_t	mean of pmc_t
pmc_t	sample mean of η_t evaluated by the Monte Carlo method
$pmc_t^{(r)}$	sample mean of η_t evaluated by the Monte Carlo method using data set r
pv_t	standard deviation of pmc_t
rpt_{ij}	random disruption in PT_{ij}
$rsut_i$	random disruption in SUT_i
rt_i	remaining time of unit agent UA$_i$ to complete the being processed task
$rtt_{ij'j}$	random disruption in $TT_{ij'j}$
sn_j	index to current task of job agent JA_j
st	current time in agent-based simulation
θ_t	vector of uncertain parameters in period t
w_{pt}	production quantity of product p in period t
x^P	vector of planning variables linked to scheduling problems
x_t^S	vector of decision variables of scheduling problem in period t
y^P	vector of planning variables other than x^P
Set	
Set_Waiting_Job	index set of jobs whose current tasks wait for processing
Set_Ready_Unit	index set of idle units ready to process a task
Set_Unit_Job	set of (i, j) such that current task of JA$_j$ can be processed in UA$_i$
Function	
η_t	function in probabilistic constraints in general formulation of integrated problem
f	objective function of upper level problem in general formulation of integrated problem
φ_t	objective function of lower level problem in period t in general formulation of integrated problem
Φ	inverse cumulative distribution function of the standard normal random variable
ψ_t	scheduling model in period t in general formulation of integrated problem
g	planning model in general formulation of integrated problem
Pr_{θ_t}	probability under uncertain parameters in θ_t

while the scheduling problem that consists of subproblems over the planning period acts as the follower. The scheduling problems in each time period are solved for their own objectives. However, their solutions (called responses in the game theory) can typically be governed by the planning problem, which is solved to optimize the entire production process. These bi-level programming methods can be regarded as a compromise between the simultaneous methods and the sequential methods. Similar to the simultaneous methods, they optimize the entire production process as an integrated problem. Moreover, these methods grant a degree of autonomy to the scheduling problems to have their own objectives like the sequential methods.

The degree of autonomy is crucial for the scheduling problems in order to deal with production uncertainties. When disruptions occur, the process is often rescheduled accordingly. The reactive scheduling decisions are commonly made autonomously according to a local objective because these decisions are made in real-time to adjust to the circumstances as they occur. Solving a complex monolithic problem to optimize a single objective under disruptions specific to scheduling in each time period is often impractical. Compared to the simultaneous methods, different objectives stemming from reactive scheduling are readily accommodated by the bi-level programming methods.

Though attractive because the model formulation aligns with the exiting organizational structure in production companies, application of the bi-level programming methods encounters some obstacles. The first challenge is computational complexity. A bi-level program is intrinsically difficult to solve. Even the simplest instance, the linear bi-level program where all problems in both upper level and lower level are linear with only continuous decision variables, has been shown to be NP-hard (Hansen, Jaumard, & Savard, 1992). Given that either a planning problem or a scheduling problem can be NP-hard (Ullman, 1975), the integrated planning and scheduling problem formulated as a bi-level program is mathematically intractable in general.

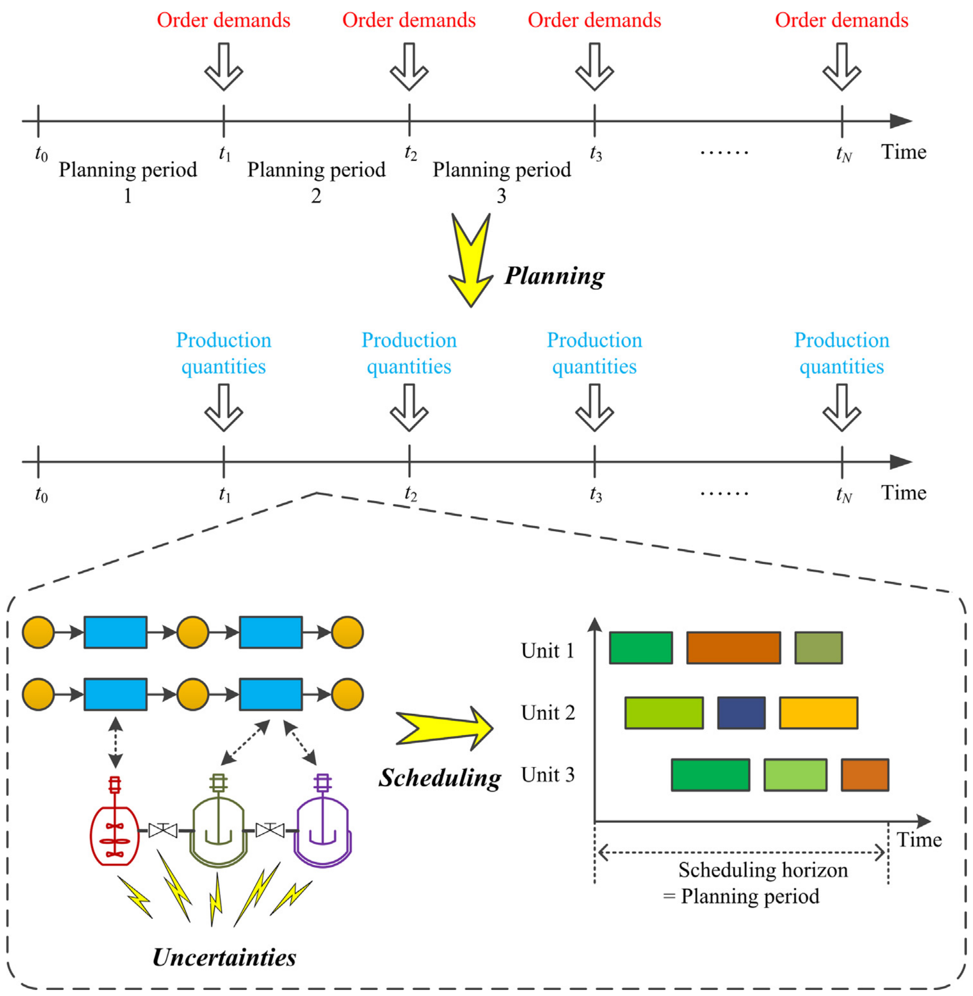


Fig. 1. The integrated planning and scheduling problem under production uncertainties.

The second difficulty in a typical bi-level programming method is the lack of capturing dynamic behaviors of the production process driven by uncertain events. When uncertainties occur in the process, scheduling decisions are not deterministic but they need to vary under different outcomes of the uncertainties. Unfortunately, the lower level problem in a common bi-level program is static and does not take the dynamic behaviors into account.

The objective of this work is to propose a new integrated method based on the bi-level model which can overcome the aforementioned challenges. The planning problem in the upper level aims to optimize the economic performance of the production process while, in the lower level, the scheduling problems over the planning periods attempt to fulfill the production plan determined by the planning problem. The planning problem is formulated as a deterministic mixed integer linear program (MILP). The reactive scheduling problems under uncertainties are solved by an agent-based method. The agent-based method can solve a large-scale scheduling problem efficiently while capturing the process uncertainties. The lower level scheduling problems are linked to the upper level planning problem through service-level constraints. The integrated problem is solved to minimize the total cost while meeting the customer order demands and satisfying the required service-levels. The hybrid problem combining the mathematical program and the agent-based modeling is solved by an iterative algorithm, in which the MILP planning problem is solved to determine the production quantities in the planning periods, while the agent-based simulation is conducted to schedule the production under uncertainties. Monte-Carlo simulations are performed to evaluate the service levels measured by probabilities. If the service levels are not satisfied, a cutting plane constraint is generated and appended to the MILP planning problem. The planning problem is solved again to determine new production quantities in the next iteration. The iterations between the MILP solver and the agent-based method are repeated until all service levels are satisfied.

Explicitly, the contribution of the paper is summarized as:

- A new formulation of the integrated planning and scheduling problem with a bi-level program and service level constraints.
- A hybrid iterative method combining an MILP solver for the planning problem with an agent-based method for the reactive scheduling problems under uncertainties.

The remainder of the paper is organized as follows. Section 2 provides the problem statement. The integrated problem is formulated in Section 3 and the iterative algorithm is developed in Section 4. The proposed method is demonstrated with the case studies in Section 5. Conclusions are given in Section 6.

2. Problem statement

The integrated problem studied in this work is shown in Fig. 1. The planning problem is the capacitated lot sizing problem, which is one of the most important and also one of the most difficult problems in production planning (Bitran & Yanasse, 1982; Karimi, Ghomi, & Wilson, 2003). The planning horizon is partitioned in a set of planning periods. We assume that the customer order demands are placed at the end of each period. The planning problem is solved to determine the production quantities for each period. The objective is to minimize the total cost that includes the inventory holding cost and the production setup cost. The determined production quantities should satisfy the order demands and the capacity of the production process so that the demanded products in every period can be delivered in that period.

Because multiple products can be manufactured in a single period, the production process needs to be scheduled in order to meet the production quantities determined by the planning model within a scheduling horizon that is equal to the length of the corresponding planning period. The inputs of the scheduling problem are the production quantities determined by the planning problem. In this work, we consider a detailed batch scheduling problem under production uncertainties. The task processing times are uncertain, which can affect the remaining unscheduled tasks. For example, a prolonged processing time can make the subsequent tasks fail to start at the predetermined time points. Because of the uncertainties, the process should be rescheduled. We adopt the reactive rescheduling policy (Relvas, Matos, Barbosa-Povoa, & Fialho, 2007; Vieira, Herrmann, & Lin, 2003) which reschedules the process according to the uncertainty outcomes.

In the integrated framework, the planning problem ensures that the production quantities assigned to all periods are fulfilled on time by solving a detailed scheduling problem for every time period. Under production uncertainties, the successful fulfillment becomes a probabilistic problem. Thus, probabilistic constraints are added to the integrated problem. We specify a threshold value, called the service level, such that the probability of the production quantities in a period to be fulfilled on time is no less than the threshold value.

Specifically, the integrated problem is stated as:

Production model
Batch process
Production uncertainties
Uncertain task processing times, sequence-dependent transition times, and unit startup times
Given
(*Planning problem*)
Planning periods
Customer order demands at the end of each planning period
Unit inventory holding cost
Unit setup cost
Thresholds of service levels
(*Scheduling problem*)
Operational tasks for manufacturing a product
Capable units for a task
Nominal task processing times
Nominal sequence-dependent transition times
Nominal unit startup times
(*Uncertainties*)
Distribution functions of uncertain parameters
Determine
Production quantities in planning periods
Reactive schedule under uncertainties for each period
Objective
To minimize total cost (= holding cost + setup cost)

3. Formulation of integrated problem

3.1. Integration framework

The integrated problem specified in the previous section is formulated as a bi-level program, as shown in Fig. 2. The "leader" in the upper level is the planning model, while the "followers" in the lower level are the scheduling models in the planning periods. The bi-level formulation is consistent with the common hierarchy in a manufacturing system, where the planning problem and the scheduling problems can be formulated with different granularities and solved with different objectives.

The planning problem determines the tactical decisions in a long horizon, like several months. It aims to optimize the economic objective while meeting the forecasted customer order demands. Because of the large time scale, the planning problem is formulated with aggregate information rather than the detailed production information. The aggregate information is used to formulate a production capacity constraint so that the planning problem does not over-assign production quantities to a planning period. In the upper tactical level, the planning problem only seeks to ensure that the determined production quantities can be fulfilled instead of how they are actually fulfilled.

The detailed production procedure is determined by the scheduling problems in the lower operational level where the detailed production information is used. The goal of a scheduling problem is to ensure that the assigned production quantities can be fulfilled by fully exploiting the production capacity. Though being influenced by the planning problem, the scheduling problems can make decisions autonomously to optimize their operational objectives which are different from the economic objective of the planning problem. This is a feature of the bi-level programming formulation that is distinct from the integrated monolithic model.

The autonomy enables a scheduling solution to immediately respond to production uncertainties without "informing" the planning module. The detailed procedure that manages uncertainties and dynamics is encapsulated inside the scheduling problems. Similar to a two-stage stochastic program (Birge & Louveaux, 2011), the scheduling decisions in the lower level depend on the uncertainty realizations, while the planning decisions in the upper level are invariant with respect to the uncertain outcomes.

3.2. Model formulation

The planning model is a capacitated lot sizing problem (Jans & Degraeve, 2008; Karimi et al., 2003), which is an MILP as below

$$\min \quad \cos t = \sum_{p,t} C_p^H inv_{pt} + \sum_{p,t} C_p^S \beta_{pt} \tag{1}$$

s.t.

$$inv_{pt} = inv_{p(t-1)} + w_{pt} - D_{pt}, \forall p, t \tag{2}$$

$$w_{pt} \le M_{pt} \beta_{pt}, \forall p, t \tag{3}$$

$$\sum_{p} A_{pt} w_{pt} \le CAP_t, \forall t \tag{4}$$

$$inv_{pt} \ge 0, \forall p, t \tag{5}$$

$$w_{pt} \ge 0, \forall p, t \tag{6}$$

$$\beta_{pt} \in \{0, 1\}, \forall p, t \tag{7}$$

The objective function (1) is the sum of the total holding cost and the setup cost, where the continuous variable inv_{pt} denotes the inventory level of product p ($p = 1, \ldots, N_p$) at the end of period t, the binary variable β_{pt} denotes if a product p is manufactured in period t, the parameters C_p^H and C_p^S represent the unit holding cost and the unit setup cost respectively. The problem includes the material balance constraint (2) where the inventory level in a period is equal to the inventory level in the previous period plus the difference between the production quantities in the period denoted by w_{pt} and the order demands denoted by D_{pt}. When the production

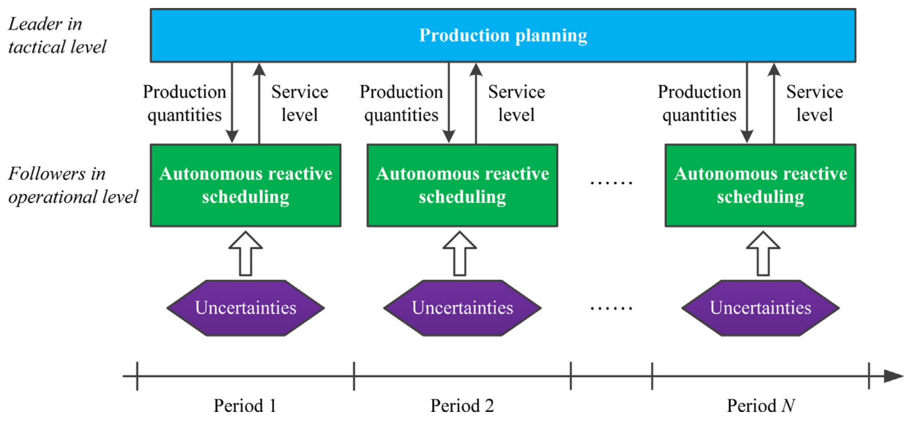


Fig. 2. Bi-level formulation of the integrated planning and scheduling problem under production uncertainties.

quantity w_{pt} is not zero, the binary variable β_{pt} is one according to the big-M constraint (3), where M_{pt} is the upper bound of w_{pt}. The production quantities are constrained by production capacities in inequality (4), where the parameter A_{pt} is the unit production time for product p in period t, and CAP_t is the maximum available production time in period t. Constraints (5)–(7) specify the feasible ranges of the decision variables.

The capacity constraint (4) uses aggregate information, which is typically a rough approximation of the real production capacity. The unit processing time A_{pt} is estimated from the detailed task processing times corresponding to the product p. However, the actual processing time of product p in period t depends on the task assignment and sequencing, which is obtained after solving a detailed scheduling problem. A rough approximation of the processing time for the parameter A_{pt} may result in infeasible production quantities for the scheduling problem (Pinedo, 2009). Moreover, approximate processing time in the capacity constraint (4) does not account for production uncertainties. As uncertainties are often described by random variables, the deterministic constraint (4) should be replaced by a probabilistic constraint. Overcoming these aforementioned shortcomings of the capacity constraint (4) is the main motivation behind developing an integrated framework in this work.

As an alternative to the constraint (4), we propose new probabilistic constraints dependent on the actual scheduling information as below

$$Pr_{\theta_t}(ms_t^*(w_{1t},\ldots,w_{pt},\ldots,w_{N_pt},\theta_t)\le H_t)\ge S_t^L,\ \forall t \tag{8}$$

where ms_t^* is the minimum makespan of the scheduling problem in period t, which depends on the production quantities w_{pt} ($p=1,\ldots,N_p$), and the uncertain parameters summarized in the vector θ_t. The operation $Pr_{\theta_t}(\cdot)$ calculates the probability, with respect to θ_t, of the makespan less than the length of the scheduling horizon (planning period), denoted by H_t. The parameter S_t^L is the given threshold of the service level. Having the makespan less than or equal to the scheduling horizon signifies successful fulfillment of the production quantities on time. The probabilistic constraints can be interpreted as that the probability of fulfilling the production quantities in a period is no less than the given service level. Although service level constraints are rarely considered in the scheduling problems, it is widely used in the process planning problems (Gupta & Maranas, 2000; Jung, Blau, Pekny, Reklaitis, & Eversdyk, 2004; You & Grossmann, 2008, 2010, 2011; Yue & You, 2013). There are different definitions for the service level. In the case studies, we consider the service level as the probability for on-time fulfillment of the production quantity in a planning period. Another common service level is defined as the percentage of the production quantities fulfilled on time in a period. For the inventory system, there are more definitions of the service level, such as the fraction of the time during which there is inventory present on the shelf and the ratio of the long-run average cumulative unsatisfied demand per replenishment cycle divided by the average demand per replenishment cycle (Goetschalckx, 2011). Because the service level constraints (8) can be replaced by a general form (12) in the following section, the proposed method can be applied to other service levels.

The minimum makespan is obtained by solving the scheduling problem for every planning period. By embedding the scheduling problems into the planning problem, the integrated problem is formulated into a bi-level program as below

$$\begin{aligned}
&(Integrated\_Problem)\quad \min\ \text{cost}\,(1)\\
&\quad \text{s.t.}\\
&\text{Planning model}\quad (2),(3),(5)\text{–}(8)\\
&ms_t^*(\{w_{pt}\}_{p=1,\ldots,N_p},\theta_t)=\min_{x_t^S}\ ms_t(\{w_{pt}\}_{p=1,\ldots,N_p},x_t^S,\theta_t),\ \forall t\\
&\quad \text{s.t.}\\
&\quad \text{Scheduling model under uncertainties in period } t
\end{aligned} \tag{9}$$

The minimum makespan is defined by the optimal value of the scheduling problem in Eq. (9), where x_t^S includes all decision variables in the scheduling problem. The scheduling problem has various formulations, e.g. the discrete-time formulation and the continuous-time formulation. Because we will use the agent-based method to solve the scheduling problem, the mathematical formulation is omitted.

4. Hybrid method

The integrated planning and scheduling problem under uncertainties formulated in the previous section is a fairly challenging problem, which could be intractable for a mathematical programming method except for a simple toy instance. For such a complex problem, a hybrid method combining mathematical programming and other approaches is more suitable (Castro, Aguirre, Zeballos, & Mendez, 2011; Harjunkoski & Grossmann, 2002; Jain & Grossmann, 2001; Li & Womer, 2009; Nikolopoulou & Ierapetritou, 2012; Roe, Papageorgiou, & Shah, 2005; Yunes, Aron, & Hooker, 2010). In this section, we develop a hybrid method combining MILP and agent-based modeling to solve the integrated problem.

4.1. Iteration framework

To facilitate the development of the solution method, we first express the integrated bi-level problem by a compact formulation as below

$$(Planning\_Problem) \quad \min_{x^P, y^P} \ f(x^P, y^P) \tag{10}$$

s.t.

$$g(x^P, y^P) \leq 0 \tag{11}$$

$$Pr_{\theta_t}(\varphi_t^*(x^P, \theta_t) \leq 0) \geq P_t, \forall t \tag{12}$$

$$(Scheduling\_Problem) \quad \varphi_t^*(x^P, \theta_t) = \min_{x_t^S} \ \varphi_t(x^P, x_t^S, \theta_t), \forall t \tag{13}$$

s.t.

$$\psi_t(x_t^S, \theta_t) \leq 0, \forall t \tag{14}$$

All variables in the planning model are stacked into two vectors denoted by x^P and y^P. The vector x^P contains all variables whose value will be used in the scheduling problems, while the vector y^P contains the remaining variables. The objective function (10) represents an economic index of the planning problem. The planning model is summarized by constraint (11). Inequalities (12) indicate the probabilistic constraints that link the planning problem to the scheduling problems. In this work, the optimal-value functions are

$$\varphi_t^*(x^P, \theta_t) = ms_t^*(\{w_{pt}\}_{p=1,\ldots,N_p}, \theta_t) - H_t, \forall t \tag{15}$$

where ms_t^* and H_t are defined in Eq. (8). The optimal-value functions are evaluated by solving the scheduling problems (13). The variables in the scheduling problem in period t are stacked into the vector x_t^S and the uncertain parameters are expressed by the vector θ_t. Constraints (14) represent the scheduling models under uncertainties. The general formulation (10)–(14) not only represents the integrated problem formulated in the previous section, but can also portray other planning and scheduling problems with different constructs. For example, the lot-sizing planning problem can be replaced by a multi-site planning problem where the production quantities are determined for different production sites instead of the planning periods.

The integrated problem (10)–(14) is extremely complicated. There are three obstacles that render it mathematically intractable except for very simple instances. First, the integrated problem is a bi-level program, which is NP-hard (Colson et al., 2007). Second, the integrated problem is subject to the service level constraints. There are often no closed-form analytical expressions for the probabilities, which are evaluated by a numerical statistical method, e.g. the Monte-Carlo method (Fu, 2002). The numerical evaluation requires solving the scheduling problems repeatedly and results in black-box functions for the service level constraints, which are difficult to handle by a common MILP solver. Third, a single deterministic scheduling problem is NP-hard (Ullman, 1975); here the integrated problem includes a number of scheduling problems over the planning periods. Furthermore, these scheduling problems are solved under uncertainties and each problem has to be solved many times to evaluate the service levels.

Exact solution for such a complicated problem is not practical for a real-world process. Therefore, we propose a heuristic method that combines an MILP solver with an agent-based method. The hybrid method is an iterative approach shown in Fig. 3. Initially, the planning problem is formulated by excluding the service level constraints. The initial problem is solved by an MILP solver to determine the production quantities and the objective function value. The objective value is a lower bound for the minimum cost, which is later used to calculate the optimality gap. Next, the determined production quantities are passed to the agent-based method,

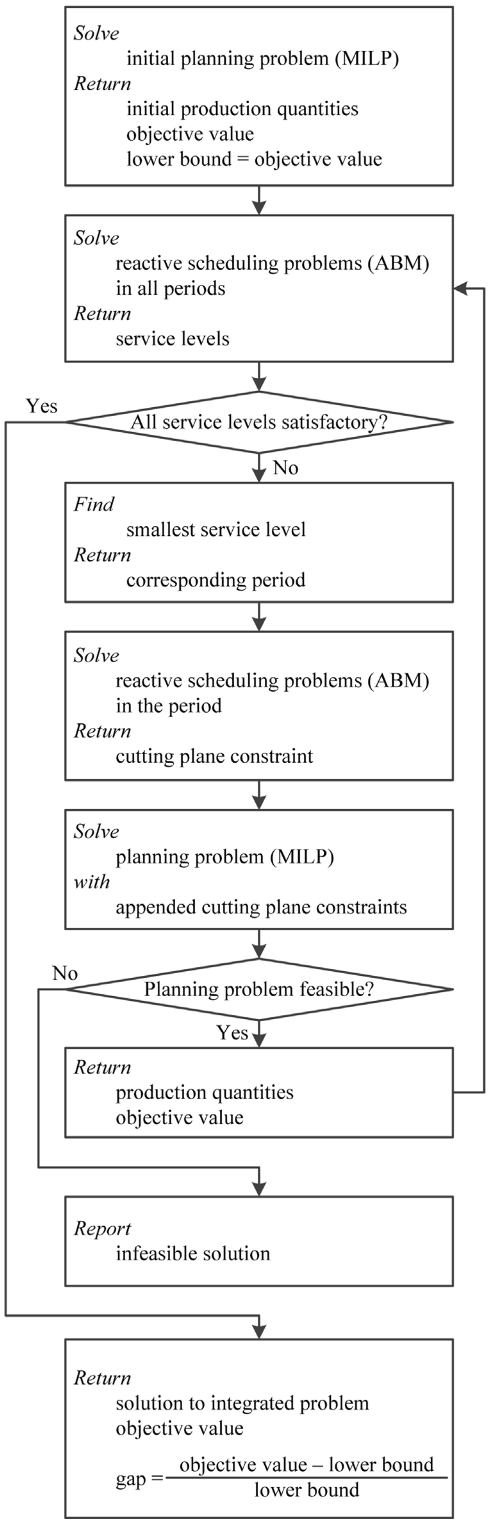

Solve
initial planning problem (MILP)
Return
initial production quantities
objective value
lower bound = objective value
Solve
reactive scheduling problems (ABM)
in all periods
Return
service levels
Yes
All service levels satisfactory?
No
Find
smallest service level
Return
corresponding period
Solve
reactive scheduling problems (ABM)
in the period
Return
cutting plane constraint
Solve
planning problem (MILP)
with
appended cutting plane constraints
No
Planning problem feasible?
Yes
Return
production quantities
objective value
Report
infeasible solution
Return
solution to integrated problem
objective value
gap = (objective value – lower bound) / lower bound

Fig. 3. Hybrid method for solving the integrated problem.

which solves the reactive scheduling problems under uncertainties over all planning periods. Monte Carlo simulation on the agent-based scheduling system is conducted to evaluate the service levels numerically. If all the service levels are satisfied, it implies that we can fulfill the production quantities assigned to all planning periods on time, and thus they are returned as the solution. Otherwise, the planning period corresponding to the smallest service level is identified. A cutting plane constraint is generated by the agent-based method by perturbing the production quantities obtained from the planning period. The cutting plane constraint is then appended to the planning problem. The updated planning problem is then solved. If the planning problem is infeasible, the integrated problem is infeasible. The algorithm terminates and reports the infeasibility. If the planning problem has the optimal solution, new production quantities are determined and the next iteration begins. The iterative procedure terminates when all services level constraints are satisfied according to the given threshold. When the algorithm terminates, it returns the objective function value, which is an upper bound of the minimum cost. By calculating the optimality gap between the upper bound returned in the last iteration and the lower bound returned in the initial iteration, we can determine the solution quality. We note that the heuristic hybrid method cannot guarantee a global optimal solution.

In this work, we focus on production uncertainties which only involve in the scheduling problems. After the integrated problem is relaxed by removing all constraints related to the scheduling problems, a deterministic planning problem is obtained. As this relaxed problem is solved exactly, the return optimal function value provides an exact lower bound for the integrated problem. Though the proposed method includes heuristics, the lower bound is valid and is not dependent on the heuristics. Similarly, the feasibility of the final solution found by the proposed method can be statistically validated in Section 4.3. In summary, though the proposed method involves heuristics, the optimality gap is not dependent on these heuristics, which provides a valid criterion to check the performance of the proposed method.

4.2. Agent-based scheduling method

Solving the planning problem by an MILP solver is straightforward. So here we focus on the agent-based scheduling method. The agent-based method determines scheduling decisions via interactions among different agents. As a distributed decision-making system, an agent-based model is adaptable to the changing environment. Therefore, the agent-based scheduling method has become increasingly popular for reactive scheduling (Shen et al., 2006; Wong, Leungy, Mak, & Fung, 2006). It can efficiently reschedule the production process, even for a large-scale problem, for observed uncertain outcomes. Besides the computational efficiency, the agent-based method can readily model various configurations in a batch scheduling problem; e.g. sequence dependent transition times, starting time dependent processing times, variable batch sizes, different storage policies, and particular scheduling rules that are a part of an industrial process (Chu, Wassick, & You, 2013; Chu, You, & Wassick, 2014).

Because it is a heuristic scheduling method, the agent-based method cannot guarantee the solution optimality. However, the solution quality of a recently developed agent-based method was shown to be very close to the optimal one by taking into account some global information along with the conventional local information (Chu et al., 2013, 2014). Actually, the goal of attaining an optimal schedule is often impractical because of the real-world complex manufacturing processes, limited computational resources, and ever-changing production environments (Ouelhadj & Petrovic, 2009). Therefore, the goal of reactive scheduling in the integrated problem is to quickly generate a schedule under uncertainties with a good performance instead of pursing the theoretically proven optimal solution.

In this work, we apply the recently developed agent-based scheduling method to a sequential batch process (Maravelias, 2012; Mendez, Cerda, Grossmann, Harjunkoski, & Fahl, 2006) where the batch integrity is preserved without splitting and mixing materials. We note that the agent-based scheduling method can be easily extended to other types of batch processes (Chu et al., 2013). Our agent-based model contains five types of agents: job agents, unit agents, a disruption agent, a scheduler agent and a timer agent. The first three types of agents are created to model the scheduling process under uncertainties, while the last two types of agents are built to implement the scheduling algorithm and conduct the simulation. The main functions of each agent are as follows.

Job agents. A job agent records the operational stages (tasks) for manufacturing a product and tracks the task execution procedure. A job agent is denoted by JA_j, with the index j. The number of job agents is determined by the production quantity of the product (#jobs for a product = production quantity/batch size). The agent-based models in different periods can have different numbers of job agents depending on the production quantities assigned in the periods. For simplicity, we assume the production quantities along with the order demands are specified directly by the number of required jobs, and they all have integer values.

Unit agents. A unit agent models a real processing unit, which is denoted by UA_i, with the index i. A unit can be used to process different tasks. When it is executing a task, the unit cannot process other tasks simultaneously. According to the common non-preemptive scheduling rule, if a unit starts executing a task, the execution will continue until the task is completed. The execution cannot be interrupted by assigning another task to the unit. The number of unit agents is equal to the number of units in the production process.

Disruption agent. The disruption agent, which is unique, generates a random number for an uncertain parameter according to its distribution and imposes the random number to the agent-based model.

Scheduler agent. The scheduler agent, which is unique, is built to implement the scheduling algorithm.

Timer agent. The timer agent, which is unique, records the simulation time and triggers the actions of other agents.

The agent-based scheduling method is illustrated in Fig. 4. The job agent JA_j records the current task waiting for execution by an attribute sn_j that indexes the task. The attribute is initialized as $sn_j = 1$, pointing to the first task of the job. When the current task is assigned to a unit, sn_j is increased by one

$$sn_j = sn_j + 1 \tag{16}$$

Then the next task becomes the current one. After the last task is processed, the job is marked completed and the job agent logs the completion time, denoted by ct_j. An uncompleted job belongs to the set

$$Set\_Waiting\_Job = \{ j \,|\, ct_j < \infty \} \tag{17}$$

where ct_j is initialized ∞ (a very large number in practice). When the job is completed, ct_j is assigned the corresponding simulation time.

The current task of an uncompleted job can be assigned to a capable unit when it is ready. A ready unit is identified via an attribute of the unit agent UA_i, denoted by rt_i, which records the remaining time to complete the task being processed in the unit. A unit is ready to process a task when the remaining time is zero, i.e. $rt_i = 0$. Thus, the set of all ready units is defined as

$$Set\_Ready\_Unit = \{ i \,|\, rt_i = 0 \} \tag{18}$$

Initially, the remaining time is set as

$$rt_i = SUT_i + rsut_i \quad \forall i \tag{19}$$

where SUT_i is the nominal startup time of UA_i and $rsut_i$ is the disruption generated by the disruption agent. The non-zero remaining times represent the preparation time before the units can start processing tasks in a period. As the simulation evolves, the remaining times get reduced (explained later in this section). Units become ready when their remaining times are reduced to zero.

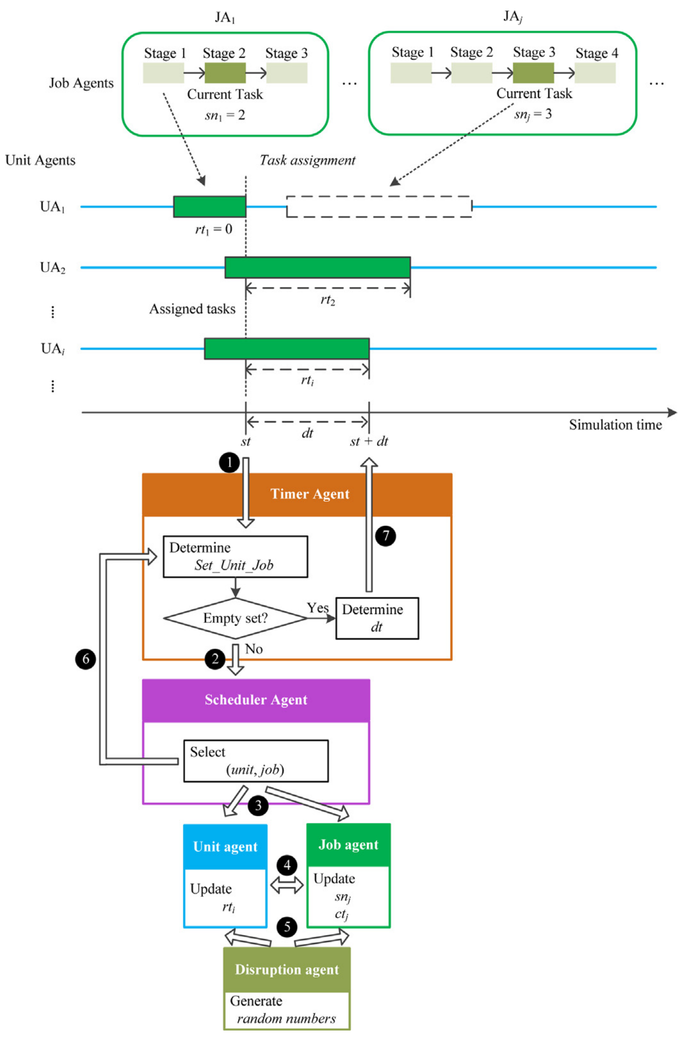


Fig. 4. Agent-based scheduling method.

The agent-based scheduling method determines the schedule by assigning waiting tasks to ready units at a simulation time point. The current simulation time is represented by *st*. The timer agent records the current simulation time and checks if task assignment can be performed at this time point. The conditions needed to assign a task require at least one waiting task and one capable ready unit, which can be stated as

$$Set\_Unit\_Job \neq \{\ \} \tag{20}$$

where the set is

$$Set\_Unit\_Job = \{(i,j) \big| \ i \in Set\_Ready\_Unit \quad \text{and} \quad j \in Set\_Waiting\_Job$$
$$\text{and} \quad UA_i \quad \text{can process the current task of} \quad JA_j\} \tag{21}$$

If *Set_Unit_Job* is not empty, task assignment can be performed. If the set contains only a pair of (i, j), the current task of JA_j is assigned to UA_i. Otherwise, the set is passed on to the scheduler agent, which selects a pair (i, j) from the *Set_Unit_Job*. The selection process can be sophisticated like a negotiation procedure (Sousa & Ramos, 1999) or a search algorithm (Chu et al., 2014). In this work, we adopt a simple rule of minimum processing time (Baker, 1998) to select the pair (i, j). A selected task JA_j has the minimum processing time when it is executed on a unit UA_j that is selected using the following rule:

$$\text{selected}(i,j) = \underset{(i,j)}{argmin} \quad PT_{ij}, \quad \text{for} \quad (i,j) \in Set\_Unit\_Job \tag{22}$$

where PT_{ij} is the nominal processing time for unit i to process the task belonging to job j.

After a pair (i, j) is selected, the current task JA_j is assigned to UA_i. The two agents communicate with each other. UA_i records the assigned task while JA_j logs the unit assigned to process the task. UA_i then updates the remaining time rt_i according to the task processing time, which can be affected by uncertainties. In the simulation, the disruption agent generates a random number and adds it to the nominal task processing time PT_{ij} and transition time $TT_{ij'j}$. The actual remaining time before the unit finishes the task becomes

$$rt_i = TT_{ij'j} + PT_{ij} + rtt_{ij'j} + rpt_{ij} \tag{23}$$

where $rtt_{ij'j}$ represents the disruption in the transition time while rpt_{ij} denotes the disruption in the processing time.

The random numbers are generated from the probability distributions of the uncertain parameters. The job agent JA_j then updates the task index sn_j according to Eq. (16). When all operational stages of a job are processed, the job agent records the completion time ct_j

$$ct_j = st + rt_i \quad \text{when} \quad JA_j \quad \text{is completed} \tag{24}$$

As defined earlier, *st* records the current simulation time.

After the task is assigned to the unit, the unit agent UA_i is removed from the set of ready units. The timer agent determines a new set of *Set_Unit_Job* according to Eq. (21). If the set is not empty, another selection of (*unit*, *job*) is made by the scheduler agent and a new task is assigned. As all these selections can be performed when the simulation time is not changed, multiple tasks can be assigned at the current simulation time *st*.

If *Set_Unit_Job* is empty, no task can be assigned. The timer agent calculates the next time point at which the tasks will be available to get assigned. The time step *dt* is calculated as

$$dt = \min_i \quad rt_i, \quad \text{for} \quad i \in \{i \big| \ \exists \quad JA_j$$
$$\text{whose current task can be processed by } UA_i\} \tag{25}$$

Then the timer agent drives the agent-based system to the next time point and the simulation time is changed to

$$st = st + dt \tag{26}$$

At the new time point, the remaining time for a unit agent becomes

$$rt_i = \max\{rt_i - dt, 0\} \tag{27}$$

When the remaining time is zero, the unit agent is put into *Set_Ready_Unit* according to Eq. (18). The scheduler agent is invoked to assign tasks. The simulation terminates when all jobs are completed and the makespan is

$$ms = \max_j \quad ct_j \quad \text{when} \quad Set\_Waiting\_Job \quad is \quad empty \tag{28}$$

4.3. Service level constraints and cutting plane method

The hybrid method iterates between the MILP solver and the agent-based scheduling method. The agent-based method can easily import the production quantities determined by the MILP solver. However, it is not straightforward for the MILP solver to use the results returned by the agent-based modeling. These agent-based models are black-box functions for the MILP solver. Though the agent-based simulation can return the outputs of the black-box functions for the given inputs, it cannot provide a closed-form expression for the input–output relationship. Therefore, we need to approximate the black-box functions by some functions with explicit expressions. The approximating functions that we adopt in this work are linear estimators generated successively. We use a cutting plane method (Atlason, Epelman, & Henderson, 2004; Cezik & L'Ecuyer, 2008; Tsai & Zheng, 2013) for simulation based optimization.

The agent-based reactive scheduling method can evaluate the objective function value

$$\varphi_t^*(x^P, \theta_t), \forall t$$

in Eq. (13). The function value depends on the planning variables x^P and the uncertain parameters θ_t. To evaluate the probabilities in the service level constraints (12), we conduct Monte-Carlo simulations. A number of sampled uncertain parameters are generated as

$$\theta_t^{(1)}, \ldots, \theta_t^{(m)}, \ldots, \theta_t^{(N_{MC})} \quad \text{from the distribution of} \quad \theta_t, \forall t \tag{29}$$

where the superscript m denotes the sample index and N_{MC} is the total number of the samples. At each sampled value $\theta_t^{(m)}$, the scheduling problem is solved by the agent-based method which returns the objective value of

$$\varphi_t^*(x^P, \theta_t^{(m)}), \forall t,\ m$$

The probability can be evaluated by the sample mean as

$$Pr_{\theta_t}(\varphi_t^*(x^P, \theta_t) \le 0) \approx \frac{\sum_{m=1}^{N_{MC}} I_t^{(m)}(x^P)}{N_{MC}}, \forall t \tag{30}$$

where the indication function is defined as

$$I_t^{(m)}(x^P) = \begin{cases} 1, \varphi_t^*(x^P, \theta_t^{(m)}) \le 0 \\ 0, \varphi_t^*(x^P, \theta_t^{(m)}) > 0 \end{cases}, \forall t, m \tag{31}$$

The Monte-Carlo method is a general computational approach which is widely used to obtain numerical solutions to problems which are too complicated to solve analytically (Pagnoncelli, Ahmed, & Shapiro, 2009; Shapiro et al., 2009). When the number of sampling points tends to infinity, the sample average on the right hand side of Eq. (30) converges to the probability value on the left hand side according to the law of large numbers (Asmussen & Glynn, 2007). However, when the sample points are finite, a discrepancy occurs between the sample average and the probability value. It cannot be 100% ensured that the probabilistic constraints (12) are satisfied when the probabilities are evaluated by the Monte Carlo method according to Eq. (30). When the Monte Carlo method is used, it is ensured that each constraint is satisfied with a given probability (Shapiro et al., 2009). Specifically, we ensure that the constraint of

$$pmc_t = \frac{\sum_{m=1}^{N_{MC}} I_t^{(m)}(x^P)}{N_{MC}} \ge P_t, \forall t \tag{32}$$

is satisfied with a given probability, denoted by P_{MC}. The sample average is denoted by pmc_t, which is a random variable dependent on the sampling points.

To test the validity of inequality (32) with the given probability P_{MC}, we need to investigate the distribution of pmc_t. When the sample size N_{MC} is large, the distribution of pmc_t converges to a normal distribution according to the central limit theorem (Asmussen & Glynn, 2007). Let pm_t denote the mean of pmc_t and pv_t denote the standard derivation. Inequality (32) is satisfied with the probability P_{MC} if

$$pc_t - \Phi(1 - P_{MC}) pv_t \ge P_t, \forall t \tag{33}$$

where the function $\Phi(\cdot)$ is the inversed cumulative distribution function of the standard normal random variable. The left hand side of Eq. (33) represents the 1–P_{MC} quantile in the distribution function of pmc_t. The probability for pcm_t to have a value larger than the quantile is P_{MC}. Therefore, if the quantile is large than the threshold value P_t. The inequality (32) is satisfied with a probability at least equal to P_{MC}. The mean pm_t and the standard deviation pv_t is estimated by repeating the Monte Carlo method over different data sets. A data set of the uncertain parameters is generated according to Eq. (29) as

$$\theta_t^{(r,1)}, \ldots, \theta_t^{(r,m)}, \ldots, \theta_t^{(r,N_{MC})} \quad \text{from the distribution of} \quad \theta_t, \forall r, t \tag{34}$$

where r is the index of the data set. Then the sample average over data set r is calculated as

$$pmc_t^{(r)} \approx \frac{\sum_{m=1}^{N_{MC}} I_t^{(r,m)}(x^P)}{N_{MC}}, \forall r, t \tag{35}$$

where $pmc_t^{(r)}$ denotes the sample average (the value returned by the Monte Carlo method) over data set r. The mean and the standard deviation of pmc_t are calculated as

$$pm_t = \frac{\sum_{r=1}^{N_R} pcm_t^{(r)}}{N_R}, \forall t \tag{36}$$

$$pv_t = \sqrt{\frac{\sum_{r=1}^{N_R} (pcm_t^{(r)} - pm_t)^2}{N_R - 1}}, \forall t \tag{37}$$

where N_R is the number of data sets.

The probability in Eq. (30), calculated using Monte Carlo simulations, is a function of the planning variables x^P. We define it as

$$\eta_t(x^P) = Pr_{\theta_t}(\varphi_t^*(x^P, \theta_t) \le 0), \forall t \tag{38}$$

As there is no close-form expression for $\eta_t(x^P)$, we bound it by the following linear function

$$\eta_t(x^P) \le \alpha_{tn}(x^P - x_n^P) + \delta_{tn}, \forall t, n \tag{39}$$

where x_n^P is the column vector of the planning variables from current and/or previous iterations, α_{tn} is the coefficient row vector, and δ_{tn} is the interception point. The index n denotes the iteration number. In this work, x^P contains integer numbers as it represents the production quantities in terms of the number of jobs. The coefficient vector in the linear function is calculated by

$$\alpha_{tn}^{(s)} = \begin{cases} \dfrac{\eta_t(x_n^P + E_s) - \eta_t(x_n^P - E_s)}{2}, & x_n^{P(s)} \ge 1 \\ \eta_t(x_n^P + E_s) - \eta_t(x_n^P), & x_n^{P(s)} = 0 \end{cases}, \forall t, n, s \tag{40}$$

where the index s in a vector denotes an element in the vector. Specifically, $\alpha_{tn}^{(s)}$ returns the sth element in α_{tn} and $x_n^{P(s)}$ indicates the s th element in x_n^P. The vector E_s is the s th column of the identity matrix. Each element in the coefficient vector α_{tn} is calculated according to Eq. (40). The value of δ_{tn} is

$$\delta_{tn} = \eta_t(x_n^P), \forall t, n \tag{41}$$

The upper bounding in inequalities (39) is valid if the continuous relaxation of the function $\eta_t(x^P)$ is concave (Atlason et al., 2004). An illustration of the upper bounding function is shown in Fig. 5. The linear bounding function passes through the point $(x_n^P, \eta_t(x_n^P))$ with the slope equal to that of the line linking two adjacent points $x_n^P - 1$ and $x_n^P + 1$ ($E_s = 1$ in this case).

Using the upper bounding function, the service level constraints (12) can be relaxed by linear constraints as

$$\alpha_{tn}(x^P - x_n^P) + \delta_{tn} \ge P_t, \forall t, n \tag{42}$$

Solving the planning problem with the linear constraints returns a lower bound of the objective value and a new value of x^P, which is denoted by x_{n+1}^P and passed to the agent-based scheduling method. As more linear constraints (42) are generated during the iteration

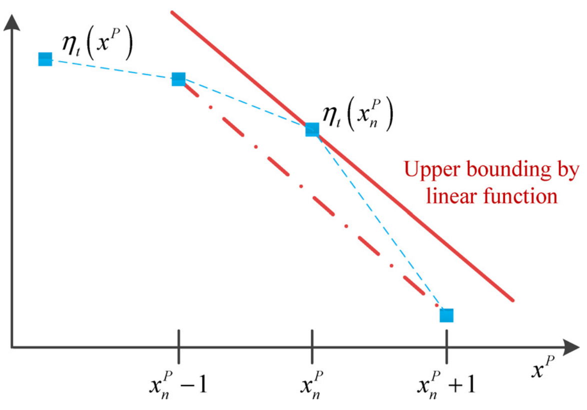


Fig. 5. Illustration of upper bounding by a linear function.

procedure shown in Fig. 3, an increasingly tight approximation is obtained. The iteration terminates when the value of x^P satisfies all service level constraints.

After the cutting plane constraints are generated, they are appended to the planning model (10)–(12) to determine the new production quantities. The planning problem with cutting plane constraints becomes

$$\min_{x^P, y^P} f(x^P, y^P)$$

s.t.

$$g(x^P, y^P) \le 0$$

$$\alpha_{tn}(x^P - x_n^P) + \delta_{tn} \ge P_t, \forall t, n$$

The original service level constraints (12) are replaced by successive cutting plane constraints (42) where n is the iteration index. The iteration procedure can be seen in Fig. 3.

The cutting plane in Eq. (39) is an upper bounding function of $\eta_t(x^P)$ when it is concave. In this case, the feasible range of $\{x^P \,|\, \eta_t(x^P) \le P_t\}$ is convex. As $\eta_t(x^P)$ evaluates the service level in period t, such service-level function is concave in the region where the service level is close to 100% (Atlason et al., 2004; Jung, Blau, Pekny, Reklaitis, & Eversdyk, 2008). Of course, the concavity of $\eta_t(x^P)$ cannot be guaranteed for all problems and the cutting plane constraints (42) may exclude some feasible values. The optimal solution may be inadvertently cut off and the cutting plane method may result in a sub-optimal solution. Because the cutting plane method is a heuristic approach, it cannot be guaranteed that this method would work well for general problems. However, we do provide a procedure for validating the performance of the cutting plane method for a specific problem. The quality of the returned solution is validated by calculating the optimality gap shown in Fig. 3.

4.4. Illustrative example for cutting plane method

We use a simple example to illustrate the cutting plane method. Our focus in this section is to explain how service level constraints are approximated by the successive linear constraints. Therefore, we solve only the scheduling problem under uncertainties in a single period to motivate the idea. Application of the entire hybrid method to a complicated planning and scheduling problem will be demonstrated using detailed case studies in the next section.

To help visualize, we consider only two jobs. Each job has two operational stages and the task in a stage can be executed in one of the two parallel units. The task processing times, the unit startup

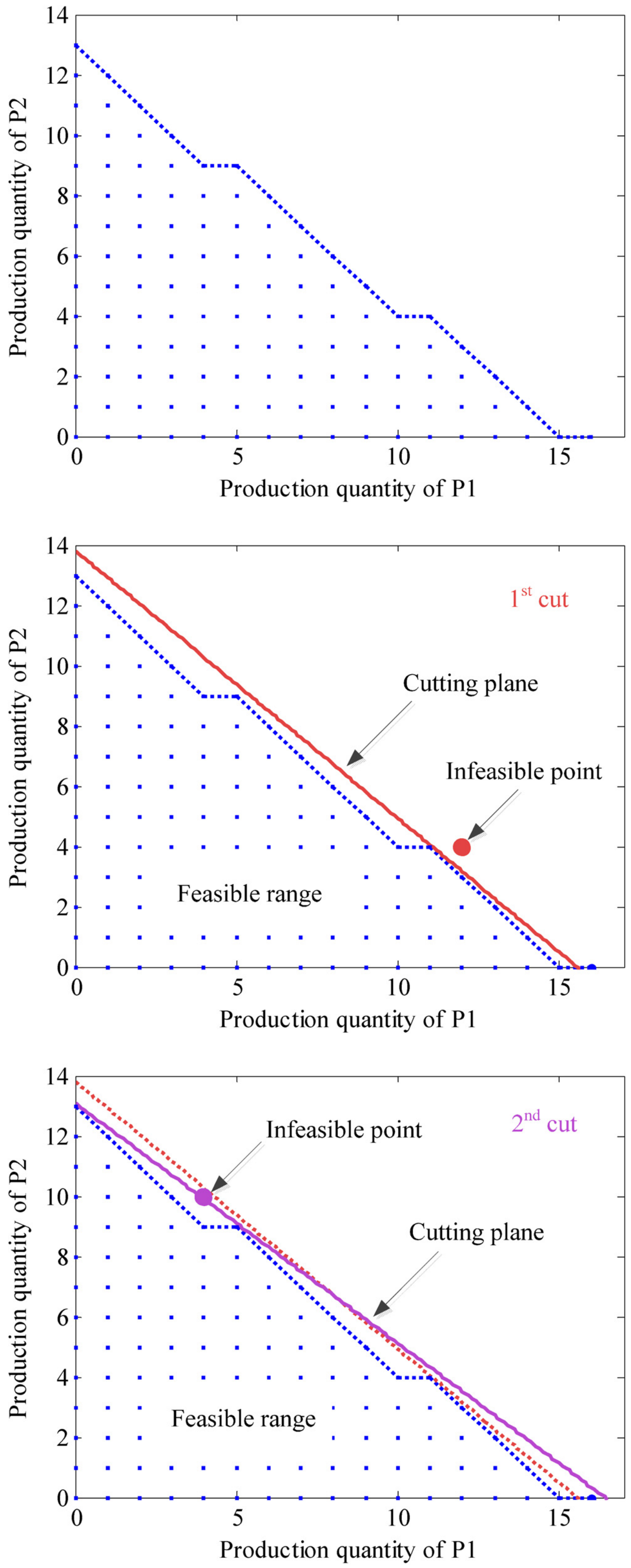


Fig. 6. Illustration of the cutting plane method.

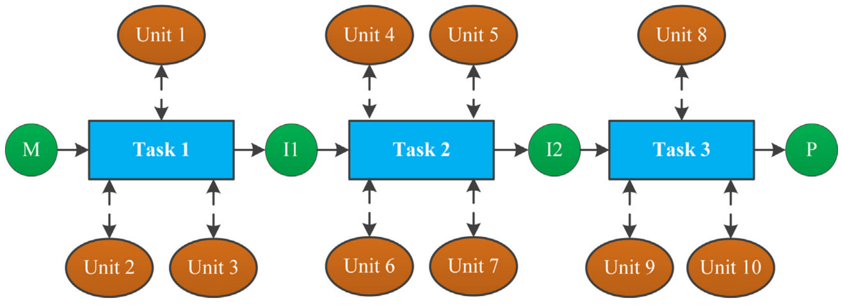


Fig. 7. Job routing diagram of case studies.

Table 1
Data of the illustrative example (minute).

	Stage 1		Stage 2		Uncertainty range
	Unit 1	Unit 2	Unit 3	Unit 4	
Job 1	80	90	70	80	±25%
Job 2	100	110	90	90	±25%
Startup time	20	20	10	10	±20%
Transition time	20	20	10	10	±20%

times, and the transition times are listed in Table 1. The times are uncertain parameters that follow a uniform distribution with the ranges given in Table 1.

The threshold of the service level is set to 0.95. We conduct 1000 simulations to sample uncertain parameters in order to evaluate the service level numerically. The production quantities for the scheduling problem take only integer values (numbers of jobs). The feasible values at which the service level is greater than or equal to 0.95 are plotted as blue points in Fig. 6(top). They were identified by an enumeration procedure. Because the continuous relaxation of $\eta_t(x^P)$, defined in Eq. (38), is not concave in this case, the continuous relaxation of the feasible range for the service level constraint is not convex. However, the feasible range is approximately a triangle, which is very close to being convex.

When an infeasible value of the production quantities is given as the red point in Fig. 6(middle), the cutting plane is generated as the red straight line. The coefficients of the cutting plane function are calculated according to the method in Section 4.3, specifically following Eqs. (40) and (41). The straight line is generated by enforcing the equality in the cutting plane constraints (42), i.e.

$$\alpha_1(x^P - x_1^P) + \delta_1 = P \tag{43}$$

where the index $n = 1$ in this example and the index t is dropped for the single period. The red point denotes x_1^P. Because x_1^P is an infeasible point, we have $\delta_1 = \eta_t(x_1^P) < P$. So the red point x_1^P is not on the red line determined by Eq. (43).

The red line provides a tight cut for the feasible range. Nevertheless, it is possible that another infeasible point can satisfy this cutting plane constraint. For example, the infeasible purple point in Fig. 6(bottom) is below the cutting plane but it is still outside the feasible range. At this point, another cutting plane is generated. The combination of the two cutting planes provides a tighter approximation for the feasible range.

As the feasible range is not convex, the cutting planes exclude some feasible points, e.g. (16, 0). This may result in a suboptimal solution for the integrated problem. However, the solution quality can be validated, which will be demonstrated in the case study.

5. Case studies

In this section, we solve a complicated planning and scheduling problem to demonstrate the hybrid method. The planning problem consists of 12 periods, each of which has a duration of 7 days (10,080 min). Customer demands are placed for 10 products (labeled by A, B, C, . . ., J). The goal of the planning problem is to determine production quantities in each period so that the customer orders can be satisfied and the total inventory cost and startup cost is minimized. As the batch sizes in this sequential batch process is fixed, the customer demands and the production quantities are quantified by numbers of jobs. For example, the customer demand for product A in period 1 is 2 jobs, the production quantity for product D in period 5 is 12 jobs, etc.

The production quantities determined in a planning period are then passed on to the scheduling problem in that period, which is then solved to determine a detailed schedule for manufacturing the demanded products. The scheduling problem includes multiple operational stages and parallel processing units. The production process of a product includes 3 operational stages. The task in each stage can be processed by one of parallel units. The job routing

Table 2
Production quantities determined in the initial iteration (Unit: #jobs. The threshold of the service level is set as 0.95).

Product	Period											
	1	2	3	4	5	6	7	8	9	10	11	12
A	2	11	0	8	16	0	11	0	9	0	8	0
B	4	6	0	3	5	12	0	4	12	0	6	0
C	3	16	0	7	0	12	14	0	15	8	11	0
D	7	0	18	0	12	0	6	11	0	8	0	11
E	6	0	7	0	5	12	0	14	0	11	0	3
F	10	0	6	15	0	5	9	0	9	0	8	0
G	15	0	13	0	11	0	17	0	5	19	0	0
H	8	5	16	0	12	13	0	6	0	9	0	0
I	4	12	0	10	0	9	0	0	8	0	16	0
J	8	0	12	0	8	0	15	0	13	0	7	0
Service level	1.00	1.00	0.37	1.00	0.75	1.00	0.27	1.00	0.57	1.00	1.00	1.00

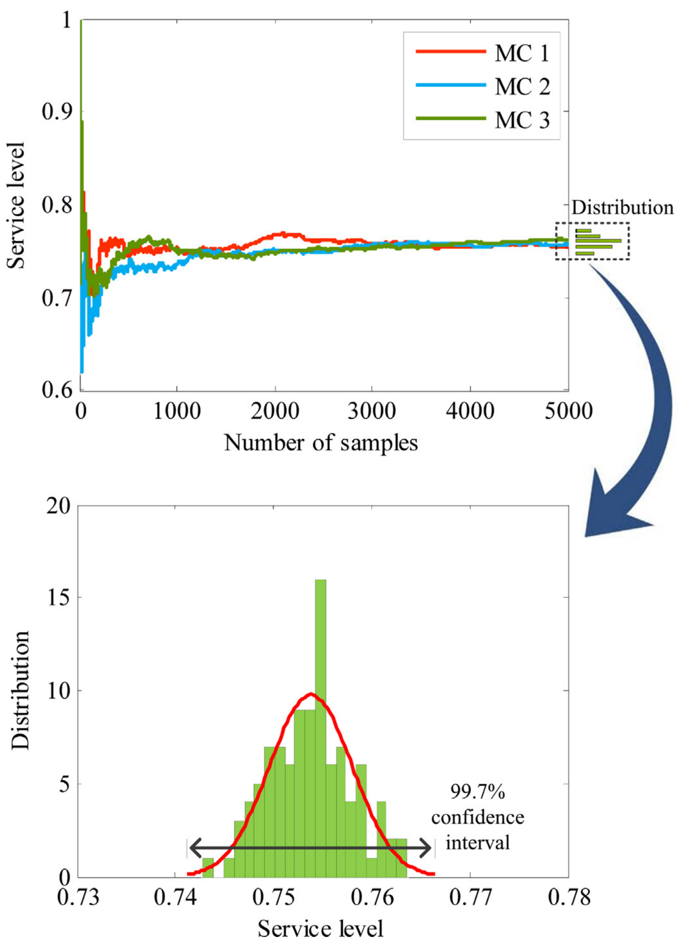


Fig. 8. Distribution of the service level for period 5 over 100 data sets. Each data set contains 5000 samples of the uncertain parameters. For clarity, the top figure only plots three Monte-Carlo simulation results according to the first three data sets. The average service level in the *y*-axis is dependent on the number of samples in the *x*-axis. The subsets with the size ranging from 1 to 5000 are selected randomly. The selection of the subsets does not affect the final average service level over the 5000 samples. The oscillation in the estimated service level reduces as more samples are used. The bottom figure shows the distribution of the service levels over the 100 data sets when all 5000 samples are used in each data set.

diagram is shown in Fig. 7. Specifically, the first task of a job can be processed by Unit 1, Unit 2, or Unit 3. The second task can be processed by Unit 4, Unit 5, Unit 6, or Unit 7. Finally, the third task can be processed by Unit 8, Unit 9, or Unit 10.

The scheduling problems are subject to production uncertainties, represented by uncertain task processing times, transition times, and unit startup times. All uncertain parameters follow a uniform distribution. The uncertainty range of the processing time is set to be ±25% of its nominal value. The uncertainty range of the transition time or the unit startup time is set to be ±20% of the nominal value. The nominal values are given in Appendix A. Appendix A also includes the data of the planning problem, such as the inventory holding costs, the production setup costs, and the customer demands.

The planning problem is modeled by GAMS 24.1.2 and solved by CPLEX 12.0 to zero optimality gap. The agent-based model is coded using Java in Eclipse (Kepler release). The MILP solver and the agent-based model exchange data using text files. We used an Intel CPU (3.10 GHz) with 4 cores and 8 GB memory for processing, and Windows 7 Professional (64 bit) as our operating system.

We should note the problems solved in the case studies are very complicated. Each integrated problem includes a planning problem and a number of detailed scheduling problems under uncertainties. Such complicated problems cannot be solved by a common method, like the mixed-integer programming method. Therefore, we do not conduct comparisons in the case studies. Instead, the performance of the proposed method is validated by the optimality gap and the statistical test presented in section 4.

5.1. Case study I (service level = 0.95)

In this subsection, we solve the integrated planning and scheduling problem by setting the threshold of the service level as 0.95. In the first iteration, the planning problem is solved without taking the scheduling problems into account. The production quantities determined for each period are shown in Table 2. The planning problem includes 373 equations, 241 continuous variables, and 960 binary variables. The problem is solved by CPLEX 12.0 in 3.9 s.

We obtained a minimum cost of 904 m.u. (money units). Note that because the scheduling problem has not been taken into account, the production quantities assigned to certain periods result in lower service levels. Particularly, the production quantities assigned to periods 3, 5 7, 9 cannot be fulfilled on time, resulting in a service level lower than 0.95. The lowest service level of 0.27 occurs in the period 7. The four unsatisfied service levels illustrate the necessity of solving an integrated planning and scheduling problem.

The service levels listed in the bottom row of Table 2 are calculated by simulating the agent-based scheduling process for each planning period 5000 times with different parameter values sampled from their uniform distributions. In one agent-based simulation, a sampled parameter vector is used to evaluate the makespan. Then the service levels are calculated by the average value over the parameter samples according to Eq. (30) where $N_{MC} = 5000$. The calculated service levels are actually random variables dependent on the data set, i.e. the 5000 parameter samples for the agent-based simulation. When another data set of 5000 samples is used, the evaluated service levels can vary. To investigate the variations of the service levels over different data sets, the service levels are calculated with 100 data sets, each having 5000 samples of the uncertain parameters (the agent-based model runs 12(#periods) × 100(#data sets) × 5000(#samples) = 6 million times in total). As an example, the distribution of the service level in period 5 is shown in Fig. 8.

The horizontal axis shows the number of samples and the vertical axis shows the service level calculated by using the number of samples. Each curve represents the result of a Monte-Carlo simulation using a data set of 5000 parameter samples. The average service level over the 100 data sets is 0.754 and the length of the 99.7% confidence interval is 0.0253, which is only 3.4% of the average value. We found the ratio of the confidence interval to the average service level to be less than 5% in all time periods. Therefore, a data set of 5000 parameter samples is adequate to obtain an accurate estimation of the service levels.

When the production quantities have been determined, they are passed on to the scheduling problem for each planning period. The production process in a period is then scheduled by the agent-based method under uncertainties. To illustrate the scheduling results, the Gantt charts of the scheduled process in planning period 5 are displayed in Fig. 9. The scheduling problem in the period is complicated, including 69 jobs and 10 units. The total number of the task bars on a Gantt chart is 207 given that each job has 3 operational stages. The unit startup times, the task processing times, and the sequence-dependent transition times are all uncertain. The single run of the agent-based scheduling model is about 0.02 s. The computational efficiency is essential for the hybrid method because the

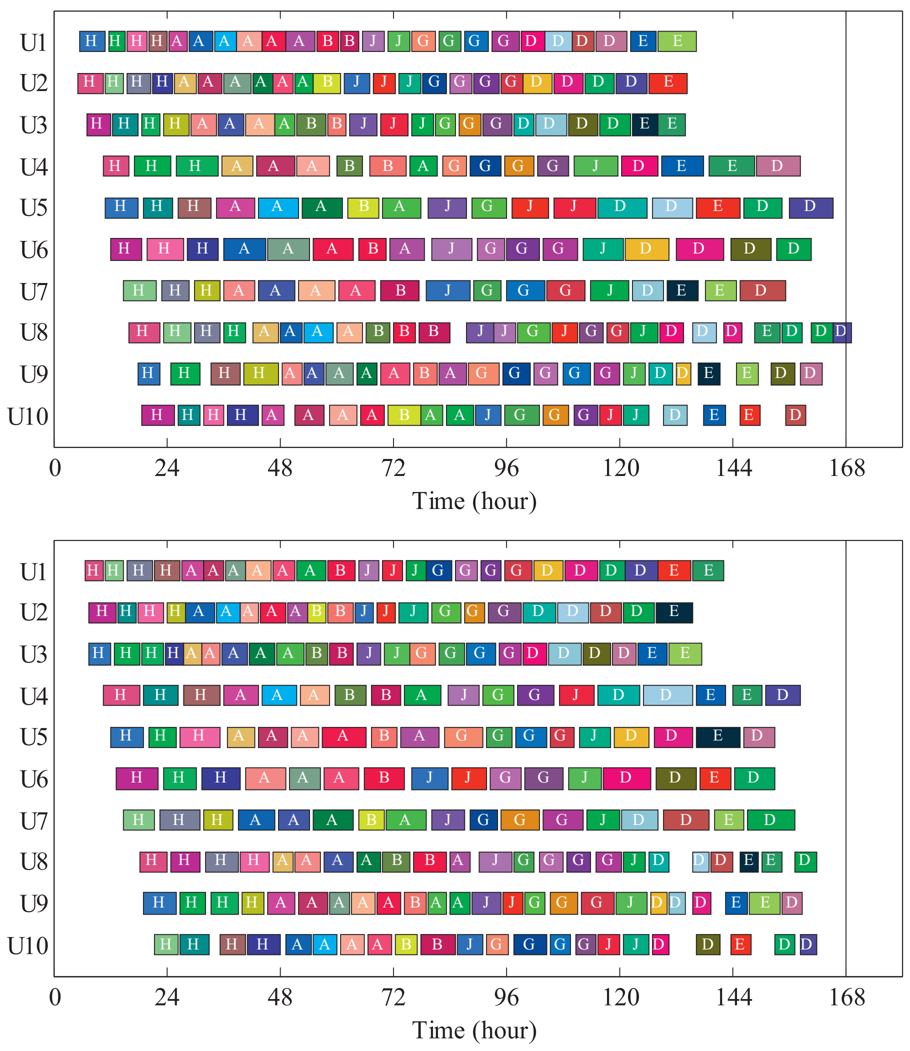


Fig. 9. Scheduling results for the planning period 5 under two different uncertain outcomes.
The top figure is an example that shows the production quantities cannot be fulfilled on time while the bottom figure is an example that shows the production quantities can be fulfilled on time. The letter on a task bar indicates the product name.

agent-based scheduling problem needs to be solved repeatedly for many times to evaluate the service levels. Under different uncertain outcomes, the scheduling results, including the task assignment and sequencing, can be different because a reactive scheduling approach is adopted. The uncertain outcomes can also affect successful fulfillment of the production quantities on time, as shown in Fig. 9.

The integrated problem is then solved by the hybrid method. The iterative results are shown in Fig. 10. In the first iteration, the service level for the planning period 7 has the lowest value of 0.27. Thus, a cutting plane constraint is generated by the agent-based simulation in the period. After the generated constraint is appended to the planning model, the production quantities get re-distributed over the planning periods. In the second iteration, the service level for the period 7 gets raised to above 0.95 because the cutting plane constraint generated in the previous iteration reduces the production quantities assigned to the period. However, the reduced production quantities in period 7 lead to increased production quantities in other periods. As a result, the service levels in periods 3, 5, and 9 are lower than those in the first iteration. In the second iteration, period 3 has the lowest service level, as shown in Fig. 10(top). Thus, a cutting plane constraint is generated by simulating the agent-based scheduling process for period 3. The same process is repeated between the optimization problem and the agent-based simulation problem. In each iteration, a cutting plane constraint is generated to raise the lowest service level in that iteration. After 8 iterations, the service levels in all planning periods were found to be above the required value of 0.95.

The evolution of the total cost along with iterations is shown in Fig. 10(middle). As the cutting plane constraints are appended to the planning problem, the optimized total cost increases. The hybrid method cannot guarantee an optimal solution. However, it is able to find an adequately good solution very close to the optimal one. The smallest cost returned by the hybrid method after the final iteration is 908 m.u., as shown in Fig. 10(middle). As all service level constraints are satisfied in the final iteration, the smallest cost returned by the hybrid method is an upper bound of the optimal cost. We note the cost in the first iteration is 904 m.u., which is the result when only the planning problem is solved. Thus a lower bound of the optimal cost is 904 m.u. The optimality gap between the smallest cost returned by the hybrid method and the lower bound is only 0.4%, demonstrating the solution quality of the hybrid method in this example.Fig. 10(bottom) exhibits the accumulative computational times after each iteration. Most

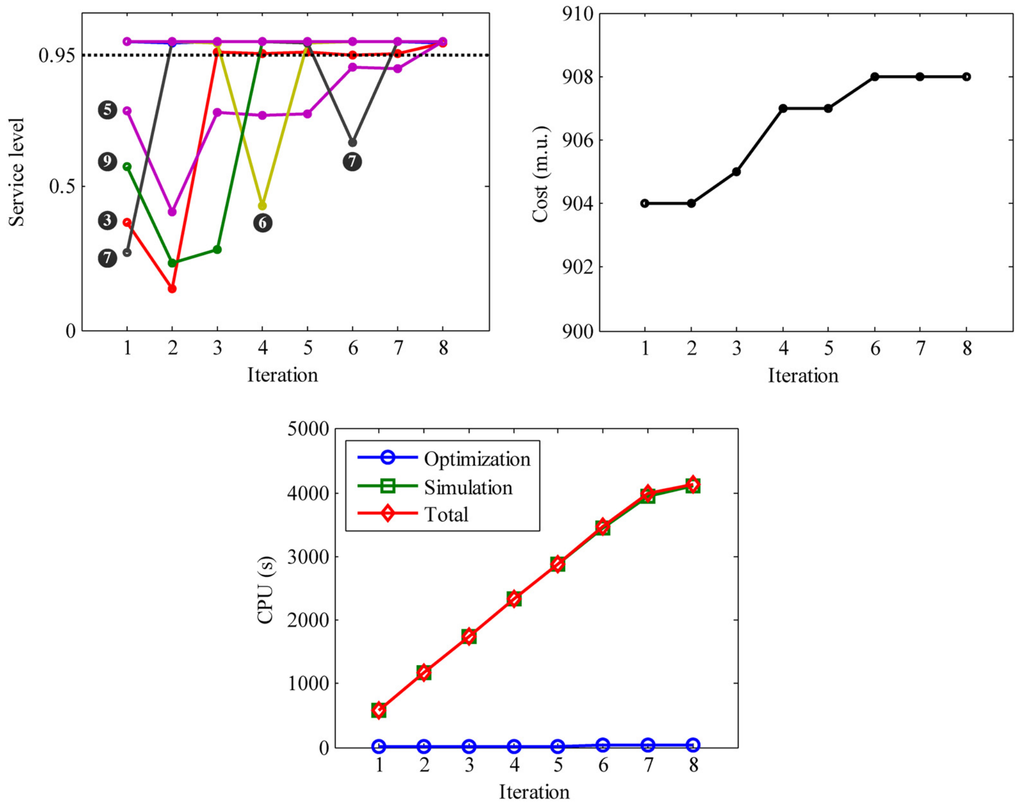


Fig. 10. Iterative results for the hybrid method (service level = 0.95).
In the top figure, the numbers in the circles denote the planning periods. In the bottom figure, the CPU times are the accumulative ones.

computational resources are spent in simulating the agent-based scheduling process for a planning period since each agent-based model runs 5000 times to calculate the service level in a period. The computational time for optimizing the planning problem is much shorter since the planning problem is solved just once in every iteration. The total computational time for all iterations is 4138.1 s (about 69 min).

The production quantities determined by the hybrid method are listed in Table 3. Compared to those in Table 2, the results

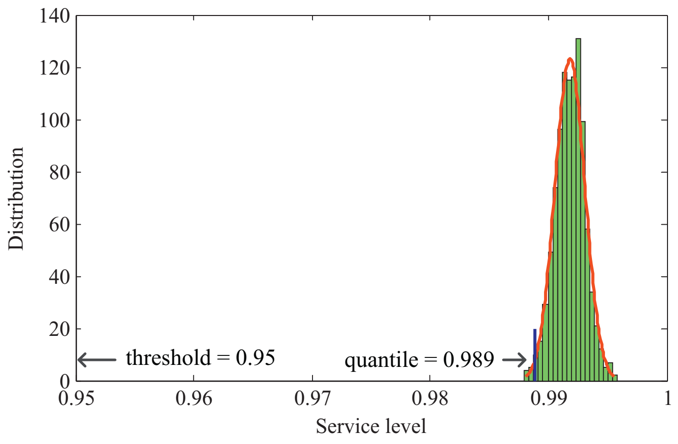
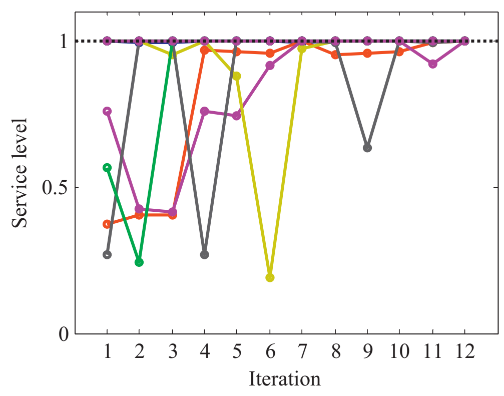


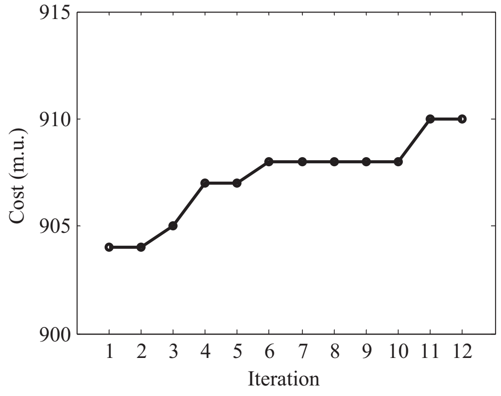
Fig. 11. Distribution of the service level in the first period for the final solution in case study I.
The 0.01 quantile is 0.989 that is larger than the threshold value of 0.95. The service level constraint in the time period is satisfied with the probability confidence of 0.99.

in Table 3 represent re-distributed production quantities over the planning periods so that the service levels in all periods are above the required value of 0.95. Because the service levels listed in Table 3 are calculated from the Monte Carlo method with a data set of 5000 samples. Their values vary for different data sets. We assume the service level constraints should be satisfied with a probability confidence equal to 0.99. To test if the solutions in Table 3 achieve the probability confidence, we calculate the service levels with 1000 data sets. The distribution of the service level in period 1 is shown in Fig. 11. Obviously, all service levels evaluated over the 1000 data sets are larger than the threshold value of 0.95. Using the statistical test presented in Section 4.3, the 0.01 (= 1–0.99) quantile is 0.989. As the quantile is larger than the threshold value, the service level in the time period is satisfied with the 99% probability confidence. We conduct the same test for the service levels in other periods and all 0.01 quantiles are larger than the threshold value. Therefore, the production quantities in Table 3 are feasible in terms of the 99% probability confidence.

5.2. *Case study II (service level = 1.00)*

In this case study, the threshold for the service level is increased to 1.00. The planning problem is initialized in the same way as the previous case. Therefore, the production quantities determined in the first iteration do not change from the values shown in Table 2. The hybrid method starts from the initial solution and terminates after 12 iterations. The optimized production quantities are shown in Table 4. The service levels for all periods meet the required value of 1.00.





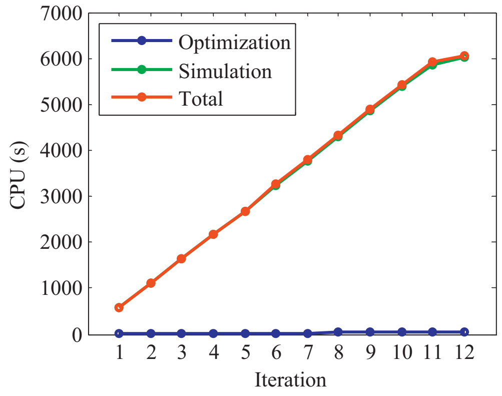


Fig. 12. Iterative results for the hybrid method (service level = 1.00).
In the bottom figure, the CPU times are the accumulative ones.

Table 3
Production quantities determined in the final iteration (Unit: #jobs. The threshold of the service level is set as 0.95).

Product	Period											
	1	2	3	4	5	6	7	8	9	10	11	12
A	2	11	0	8	16	0	11	0	9	0	8	0
B	4	6	0	3	5	12	0	4	12	0	6	0
C	3	16	0	7	0	19	0	7	15	8	11	0
D	7	0	18	0	12	0	6	11	0	8	0	11
E	6	0	7	0	5	7	5	14	0	11	0	3
F	5	11	0	15	0	5	9	0	9	0	8	0
G	15	0	7	12	0	5	17	0	5	19	0	0
H	8	5	16	0	12	13	0	6	0	9	0	0
I	10	0	10	0	12	0	0	11	0	0	16	0
J	8	0	7	13	0	0	15	0	13	0	7	0
Service level	0.99	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Table 4
Production quantities determined in the final iteration (Unit: #jobs. The threshold of the service level is set as 1.00).

Product	Period											
	1	2	3	4	5	6	7	8	9	10	11	12
A	2	4	3	7	6	5	15	8	10	3	2	4
B	11	6	16	0	0	11	0	5	0	12	11	6
C	0	0	0	18	7	0	7	16	10	0	0	0
D	8	3	7	0	0	15	12	0	0	13	8	3
E	16	5	0	12	5	0	0	12	12	0	16	5
F	0	12	19	0	7	5	5	13	0	0	0	12
G	11	0	0	6	5	9	17	0	0	15	11	0
H	0	4	7	11	14	0	0	6	11	0	0	4
I	9	12	15	0	0	9	5	0	0	13	9	12
J	0	0	8	8	11	0	19	9	0	0	0	0
Service level	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

The iteration procedure is shown in Fig. 12. The cost returned by the hybrid method when all service level constraints are satisfied is 910 m.u. The optimality gap is 0.66%. The total computational time is 6061.9 s (101.0 min). Similar to the previous case study, the agent-based simulation accounts for most of the computational time as 5000 Monte-Carlo simulations are conducted to evaluate the probability functions.

6. Conclusion

We propose a bi-level model for formulating an integrated planning and scheduling problem under production uncertainties. The upper level problem is the planning problem formulated as an MILP, while the lower level problem consists of scheduling problems over the planning periods, which are modeled by agent-based systems. The planning problem is linked to the scheduling problems via service level constraints that are probabilistic constraints dependent on detailed schedules. To solve the integrated problem, we develop a hybrid method combining an MILP solver and the agent-based method. The MILP planning problem is solved to determine production quantities over each planning period, which are then passed on to the agent-based simulations for scheduling. The agent-based method solves the reactive scheduling problems under certainties. When the service levels are not satisfied, because of the assigned production quantities exceeding the available capacity, the agent-based method generates cutting plane constraints. These constraints are then appended to the planning problem, which is then solved again to determine the new production quantities in the next iteration. The algorithm terminates when all service level constraints are satisfied.

The bi-level formulation and the hybrid method are demonstrated by two complicated case studies. In the first case study where the threshold of the service levels is set as 0.95, the hybrid method solves the problem using 8 iterations in 69 min. In the second case study where the threshold of the service levels is set as 1.00, the hybrid method solves the problem using 12 iterations in 101 min. The optimality gap for both case studies is less than 1%.

As the integrated planning and scheduling problem under uncertainties is too complicated to solve exactly, the proposed method include heuristics in the cutting plane approach and the evaluation of the probability service level functions. Also because of the complexity and the variety of the integrated problems, it is difficult to determine, for a general problem, if the cutting plane method will return a close-to-optimal solution or if the number of samples for the Monte Carlo method is adequate with a given probability confidence. However, we can validate the performance of the proposed method by applying it to a given specific problem. From the statistical test on the final solution, we can validate if the service level constraints are satisfied with the given probability confidence. From the optimality gap, we can validate if the returned objective function value is close to the optimal one.

Appendix A. Data for case studies

See Tables A.1–A.5

Table A.4
Task processing times (minute).

Product	Stage 1	Stage 2	Stage 3
A	300	450	320
B	310	420	350
C	350	500	310
D	350	520	270
E	400	490	310
F	250	420	330
G	340	410	350
H	280	440	360
I	260	430	310
J	310	450	330

Table A.1
Unit cost in planning problem (m.u.).

Product	A	B	C	D	E	F	G	H	I	J
Holding cost	1	2	1	1	2	1	1	2	1	1
Setup cost	10	9	9	11	10	7	8	10	9	12

Table A.2
Customer order demands (#jobs).

Product	Period											
	1	2	3	4	5	6	7	8	9	10	11	12
A	2	6	5	8	11	5	6	5	6	3	5	3
B	4	4	2	3	5	8	4	4	9	3	4	2
C	3	10	6	5	2	12	7	7	15	8	9	2
D	3	4	13	5	7	5	6	8	3	4	4	11
E	3	3	4	3	5	7	5	9	5	9	2	3
F	5	5	6	12	3	5	6	3	4	5	6	2
G	11	4	7	6	6	5	14	3	5	12	5	2
H	8	5	13	3	12	10	3	4	2	4	3	2
I	4	6	6	4	6	4	2	3	5	3	10	6
J	2	6	5	8	11	5	6	5	6	3	5	3

Table A.3
Unit startup times (minute).

Unit	1	2	3	4	5	6	7	8	9	10
Time	400	400	400	400	400	400	400	400	400	400

Table A.5
Sequence-dependent transition times (10 min).

Unit	Transition times
1	{A: 2,2,5,4,4,5,4,4,2,2} {B: 3,3,2,2,3,3,5,4,2,5}, {C: 5,4,3,4,4,4,4,2,5,2}, {D: 2,4,5,2,3,5,2,5,4,2}, {E: 2,5,2,2,3,5,4,2,3,5}, {F: 2,2,5,3,5,3,3,3,2,5}, {G: 3,5,5,3,2,4,4,3,5,2}, {H: 2,2,5,2,4,3,4,3,4,4}, {I: 4,2,2,5,4,5,5,2,3,3}, {J: 5,4,3,2,2,5,3,2,2,5}
2	{A:2,2,5,4,4,5,4,4,2,2}, {B:3,3,2,2,3,3,5,4,2,5}, {C:5,4,3,4,4,4,4,2,5,2}, {D:2,4,5,2,3,5,2,5,4,2}, {E:2,5,2,2,3,5,4,2,3,5}, {F:2,2,5,3,5,3,3,3,2,5}, {G:3,5,5,3,2,4,4,3,5,2}, {H:2,2,5,2,4,3,4,3,4,4}, {I:4,2,2,5,4,5,5,2,3,3}, {J:5,4,3,2,2,5,3,2,2,5}
3	{A:2,2,5,4,4,5,4,4,2,2}, {B:3,3,2,2,3,3,5,4,2,5}, {C:5,4,3,4,4,4,4,2,5,2}, {D:2,4,5,2,3,5,2,5,4,2}, {E:2,5,2,2,3,5,4,2,3,5}, {F:2,2,5,3,5,3,3,3,2,5}, {G:3,5,5,3,2,4,4,3,5,2}, {H:2,2,5,2,4,3,4,3,4,4}, {I:4,2,2,5,4,5,5,2,3,3}, {J:5,4,3,2,2,5,3,2,2,5}
4	{A:5,7,10,7,9,5,6,8,9,10}, {B:5,9,5,5,9,5,5,8,5,10}, {C:10,8,7,7,10,10,10,5,7,8}, {D:6,9,6,8,5,7,10,7,5,10}, {E:9,5,7,5,9,5,7,7,8,10}, {F:8,10,9,10,6,8,10,5,7,8}, {G:6,9,7,7,9,8,5,5,7,8}, {H:7,5,9,8,6,7,8,7,9,9}, {I:7,5,8,5,8,7,5,7,6,9}, {J:7,5,6,5,6,7,7,7,7,6}
5	{A:5,7,10,7,9,5,6,8,9,10}, {B:5,9,5,5,9,5,5,8,5,10}, {C:10,8,7,7,10,10,10,5,7,8}, {D:6,9,6,8,5,7,10,7,5,10}, {E:9,5,7,5,9,5,7,7,8,10}, {F:8,10,9,10,6,8,10,5,7,8}, {G:6,9,7,7,9,8,5,5,7,8}, {H:7,5,9,8,6,7,8,7,9,9}, {I:7,5,8,5,8,7,5,7,6,9}, {J:7,5,6,5,6,7,7,7,7,6}
6	{A:5,7,10,7,9,5,6,8,9,10}, {B:5,9,5,5,9,5,5,8,5,10}, {C:10,8,7,7,10,10,10,5,7,8}, {D:6,9,6,8,5,7,10,7,5,10}, {E:9,5,7,5,9,5,7,7,8,10}, {F:8,10,9,10,6,8,10,5,7,8}, {G:6,9,7,7,9,8,5,5,7,8}, {H:7,5,9,8,6,7,8,7,9,9}, {I:7,5,8,5,8,7,5,7,6,9}, {J:7,5,6,5,6,7,7,7,7,6}
7	{A:5,7,10,7,9,5,6,8,9,10}, {B:5,9,5,5,9,5,5,8,5,10}, {C:10,8,7,7,10,10,10,5,7,8}, {D:6,9,6,8,5,7,10,7,5,10}, {E:9,5,7,5,9,5,7,7,8,10}, {F:8,10,9,10,6,8,10,5,7,8}, {G:6,9,7,7,9,8,5,5,7,8}, {H:7,5,9,8,6,7,8,7,9,9}, {I:7,5,8,5,8,7,5,7,6,9}, {J:7,5,6,5,6,7,7,7,7,6}
8	{A:4,5,6,5,4,4,3,4,4,3}, {B:3,5,5,5,4,3,6,5,4,4}, {C:4,3,3,6,5,5,5,3,5,6}, {D:3,5,6,3,5,4,4,3,5,6}, {E:6,4,3,4,3,4,6,3,4,5}, {F:3,5,6,3,3,4,6,4,4,3}, {G:6,3,3,3,4,6,6,5,6,3}, {H:6,6,6,3,4,3,5,5,4,6}, {I:5,3,4,4,5,6,4,4,4,5}, {J:5,5,5,5,3,3,4,6,6,3}
9	{A:4,5,6,5,4,4,3,4,4,3}, {B:3,5,5,5,4,3,6,5,4,4}, {C:4,3,3,6,5,5,5,3,5,6}, {D:3,5,6,3,5,4,4,3,5,6}, {E:6,4,3,4,3,4,6,3,4,5}, {F:3,5,6,3,3,4,6,4,4,3}, {G:6,3,3,3,4,6,6,5,6,3}, {H:6,6,6,3,4,3,5,5,4,6}, {I:5,3,4,4,5,6,4,4,4,5}, {J:5,5,5,5,3,3,4,6,6,3}
10	{A:4,5,6,5,4,4,3,4,4,3}, {B:3,5,5,5,4,3,6,5,4,4}, {C:4,3,3,6,5,5,5,3,5,6}, {D:3,5,6,3,5,4,4,3,5,6}, {E:6,4,3,4,3,4,6,3,4,5}, {F:3,5,6,3,3,4,6,4,4,3}, {G:6,3,3,3,4,6,6,5,6,3}, {H:6,6,6,3,4,3,5,5,4,6}, {I:5,3,4,4,5,6,4,4,4,5}, {J:5,5,5,5,3,3,4,6,6,3}

To save space, the transition times in a unit are organized in a compact form. The transition times from a product are grouped inside the curly braces. For example, the expression "{A:2,2,5,4,4,5,4,4,2,2}" in the first row summarizes the transition times from product A in Unit 1. The transition times are ordered according to the product sequence from A to J. For example, the 1st element "2" in "{A:2,2,5,4,4,5,4,4,2,2}" represents the transition time from product A to product A (20 min). The 6th element "5" represents the transition time from product A to product F (50 min).

References

Asmussen, S., & Glynn, P. W. (2007). . *Stochastic simulation: Algorithms and analysis* (Vol. 57) New York, NY, USA: Springer.

Atlason, J., Epelman, M. A., & Henderson, S. G. (2004). Call center staffing with simulation and cutting plane methods. *Annals of Operations Research*, *127*, 333–358.

Baker, A. D. (1998). A survey of factory control algorithms that can be implemented in a multi-agent heterarchy: Dispatching, scheduling, and pull. *Journal of Manufacturing Systems*, *17*, 297–320.

Birewar, D. B., & Grossmann, I. E. (1990). Simultaneous production planning and scheduling in multiproduct batch plants. *Industrial & Engineering Chemistry Research*, *29*, 570–580.

Birge, J. R., & Louveaux, F. (2011). *Introduction to stochastic programming* (2nd ed.). New York: Springer.

Bitran, G. R., & Yanasse, H. H. (1982). Computational complexity of the capacitated lot size problem. *Management Science*, *28*, 1174–1186.

Castro, P. M., Aguirre, A. M., Zeballos, L. J., & Mendez, C. A. (2011). Hybrid mathematical programming discrete-event simulation approach for large-scale scheduling problems. *Industrial & Engineering Chemistry Research*, *50*, 10665–10680.

Cezik, M. T., & L'Ecuyer, P. (2008). Staffing multiskill call centers via linear programming and simulation. *Management Science*, *54*, 310–323.

Chu, Y., Wassick, J. M., & You, F. (2013). Efficient scheduling method of complex batch processes with general network structure via agent-based modeling. *Aiche Journal*, *59*, 2884–2906.

Chu, Y., You, F., & Wassick, J. M. (2014). Hybrid method integrating agent-based modeling and heuristic tree search for scheduling of complex batch processes. *Computers & Chemical Engineering*, *60*, 277–296.

Colson, B., Marcotte, P., & Savard, G. (2007). An overview of bilevel optimization. *Annals of Operations Research*, *153*, 235–256.

Fu, M. C. (2002). Optimization for simulation: Theory vs. practice. *Informs Journal on Computing*, *14*, 192–215.

Goetschalckx, M. (2011). . *Supply chain engineering* (Vol. 161) New York: Springer.

Gupta, A., & Maranas, C. D. (2000). A two-stage modeling arid solution framework for multisite midterm planning under demand uncertainty. *Industrial & Engineering Chemistry Research*, *39*, 3799–3813.

Hansen, P., Jaumard, B., & Savard, G. (1992). New branch-and-bound rules for linear bilevel programming. *Siam Journal on Scientific and Statistical Computing*, *13*, 1194–1217.

Harjunkoski, I., & Grossmann, I. E. (2002). Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods. *Computers & Chemical Engineering*, *26*, 1533–1552.

Jain, V., & Grossmann, I. E. (2001). Algorithms for hybrid MILP/CP models for a class of optimization problems. *Informs Journal on Computing*, *13*, 258–276.

Jans, R., & Degraeve, Z. (2008). Modeling industrial lot sizing problems: A review. *International Journal of Production Research*, *46*, 1619–1643.

Jung, J. Y., Blau, G., Pekny, J. F., Reklaitis, G., & Eversdyk, D. (2008). Integrated safety stock management for multi-stage supply chains under production capacity constraints. *Computers & Chemical Engineering*, *32*, 2570–2581.

Jung, J. Y., Blau, G., Pekny, J. F., Reklaitis, G. V., & Eversdyk, D. (2004). A simulation based optimization approach to supply chain management under demand uncertainty. *Computers & Chemical Engineering*, *28*, 2087–2106.

Karimi, B., Ghomi, S., & Wilson, J. M. (2003). The capacitated lot sizing problem: A review of models and algorithms. *Omega-International Journal of Management Science*, *31*, 365–378.

Li, H. T., & Womer, K. (2009). Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm. *Journal of Scheduling*, *12*, 281–298.

Li, Z., & Ierapetritou, M. G. (2009). Integrated production planning and scheduling using a decomposition framework. *Chemical Engineering Science*, *64*, 3585–3597.

Maravelias, C. T. (2012). General framework and modeling approach classification for chemical production scheduling. *Aiche Journal*, *58*, 1812–1828.

Maravelias, C. T., & Sung, C. (2009). Integration of production planning and scheduling: Overview, challenges and opportunities. *Computers & Chemical Engineering*, *33*, 1919–1930.

Mendez, C. A., Cerda, J., Grossmann, I. E., Harjunkoski, I., & Fahl, M. (2006). State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & Chemical Engineering*, *30*, 913–946.

Nikolopoulou, A., & Ierapetritou, M. G. (2012). Hybrid simulation based optimization approach for supply chain management. *Computers & Chemical Engineering*, *47*, 183–193.

Ouelhadj, D., & Petrovic, S. (2009). A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling*, *12*, 417–431.

Pagnoncelli, B. K., Ahmed, S., & Shapiro, A. (2009). Sample average approximation method for chance constrained programming: Theory and applications. *Journal of Optimization Theory and Applications*, *142*, 399–416.

Phanden, R. K., Jain, A., & Verma, R. (2011). Integration of process planning and scheduling: A state-of-the-art review. *International Journal of Computer Integrated Manufacturing*, *24*, 517–534.

Pinedo, M. L. (2009). *Planning and scheduling in manufacturing and services: Introduction* (2nd ed.).

Relvas, S., Matos, H. A., Barbosa-Povoa, A., & Fialho, J. (2007). Reactive scheduling framework for a multiproduct pipeline with inventory management. *Industrial & Engineering Chemistry Research*, *46*, 5659–5672.

Roe, B., Papageorgiou, L. G., & Shah, N. (2005). A hybrid MILP/CLP algorithm for multipurpose batch process scheduling. *Computers & Chemical Engineering*, *29*, 1277–1291.

Ryu, J.-H., Dua, V., & Pistikopoulos, E. N. (2004). A bilevel programming framework for enterprise-wide process networks under uncertainty. *Computers & Chemical Engineering*, *28*, 1121–1129.

Shao, X. Y., Li, X. Y., Gao, L., & Zhang, C. Y. (2009). Integration of process planning and scheduling—A modified genetic algorithm-based approach. *Computers & Operations Research*, *36*, 2082–2096.

Shapiro, A., Dentcheva, D., & Ruszczynski, A. (2009). . *Lectures on stochastic programming: Modeling and theory* (Vol. 9) Philadelphia: Siam.

Shen, W. M., Wang, L. H., & Hao, Q. (2006). Agent-based distributed manufacturing process planning and scheduling: A state-of-the-art survey. *IEEE Transactions on Systems Man and Cybernetics Part C: Applications and Reviews*, *36*, 563–577.

Shobrys, D. E., & White, D. C. (2002). Planning, scheduling and control systems: Why cannot they work together. *Computers & Chemical Engineering*, *26*, 149–160.

Sousa, P., & Ramos, C. (1999). A distributed architecture and negotiation protocol for scheduling in manufacturing systems. *Computers in Industry*, *38*, 103–113.

Tan, W., & Khoshnevis, B. (2000). Integration of process planning and scheduling—A review. *Journal of Intelligent Manufacturing*, *11*, 51–63.

Tsai, S. C., & Zheng, Y. X. (2013). A simulation optimization approach for a two-echelon inventory system with service level constraints. *European Journal of Operational Research*, *229*, 364–374.

Ullman, J. D. (1975). NP-complete scheduling problems. *Journal of Computer and System Sciences*, *10*, 384–393.

Vieira, G. E., Herrmann, J. W., & Lin, E. (2003). Rescheduling manufacturing systems: A framework of strategies, policies, and methods. *Journal of Scheduling*, *6*, 39–62.

Wassick, J. M., Agarwal, A., Akiya, N., Ferrio, J., Bury, S., & You, F. Q. (2012). Addressing the operational challenges in the development, manufacture, and supply of advanced materials and performance products. *Computers & Chemical Engineering*, *47*, 157–169.

Wong, T. N., Leungy, C. W., Mak, K. L., & Fung, R. Y. K. (2006). Integrated process planning and scheduling/rescheduling—An agent-based approach. *International Journal of Production Research*, *44*, 3627–3655.

You, F., & Grossmann, I. E. (2008). Mixed-Integer Nonlinear Programming Models and Algorithms for Large-Scale Supply Chain Design with Stochastic Inventory Management. *Industrial & Engineering Chemistry Research*, *47*, 7802–7817.

You, F. Q., & Grossmann, I. E. (2010). Integrated multi-echelon supply chain design with inventories under uncertainty: MINLP models, computational strategies. *AICHE Journal*, *56*, 419–440.

You, F. Q., & Grossmann, I. E. (2011). Stochastic inventory management for tactical process planning under uncertainties: MINLP models and algorithms. *AICHE Journal*, *57*, 1250–1277.

Yue, D. J., & You, F. Q. (2013). Planning and scheduling of flexible process networks under uncertainty with stochastic inventory: MINLP models and algorithm. *AICHE Journal*, *59*, 1511–1532.

Yunes, T., Aron, I. D., & Hooker, J. N. (2010). An integrated solver for optimization problems. *Operations Research*, *58*, 342–356.