Chiang, Nai-Yuan; Lin, Yiqing; Long, Quan

Article

Efficient propagation of uncertainties in manufacturing supply chains: Time buckets, L-leap, and multilevel Monte Carlo methods

Operations Research Perspectives

Provided in Cooperation with:
Elsevier

Suggested Citation: Chiang, Nai-Yuan; Lin, Yiqing; Long, Quan (2020) : Efficient propagation of uncertainties in manufacturing supply chains: Time buckets, L-leap, and multilevel Monte Carlo methods, Operations Research Perspectives, ISSN 2214-7160, Elsevier, Amsterdam, Vol. 7, pp. 1-15,
https://doi.org/10.1016/j.orp.2020.100144

This Version is available at:
https://hdl.handle.net/10419/246416

Operations Research Perspectives 7 (2020) 100144

Contents lists available at ScienceDirect

Operations Research Perspectives

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

Efficient propagation of uncertainties in manufacturing supply chains: Time buckets, L-leap, and multilevel Monte Carlo methods

Nai-Yuan Chiang, Yiqing Lin, Quan Long*

United Technologies Research Center, 411 Silver Lane, East Hartford, CT, USA

ARTICLE INFO

Keywords:
Uncertainty modeling
Discrete event simulation
Multilevel Monte Carlo
L-leap
Supply chain

ABSTRACT

Uncertainty propagation of large-scale discrete supply chains can be prohibitive when numerous events occur during the simulated period and when discrete-event simulations (DES) are costly. We present a time-bucket method to approximate and accelerate the DES of supply chains. Its stochastic version, which we call the L (logistic)-leap method, can be viewed as an extension of the leap methods (e.g., τ-leap [36]and D-leap [6] developed in the chemical engineering community for the acceleration of stochastic DES of chemical reactions). The L-leap method instantaneously updates the system state vector at discrete time points, and the production rates and policies of a supply chain are assumed to be stationary during each time bucket. We propose using the multilevel Monte Carlo (MLMC) method to efficiently propagate the uncertainties in a supply chain network, where the levels are naturally defined by the sizes of the time buckets of the simulations. We demonstrate the efficiency and accuracy of our methods using four numerical examples derived from a real-world manufacturing material flow application. In these examples, our multilevel L-leap approach can be faster than the standard Monte Carlo (MC) method by one or two orders of magnitude without compromising accuracy.

1. Introduction

Supply chains are coordinated flows of materials from the suppliers to the locations where they are consumed. Their dynamics can be very complex due to heterogeneity and multi-scale phenomena. In manufacturing supply chains, resources are consumed, and products are manufactured in the factories, parts are transported from factories to warehouses, and final products are delivered to customers. Various policies, for example, safety stock control [11,49], and push and pull systems [58,59], can be designed to manage the rate of production and the supply of inventory. Simulation is a powerful tool to model the dynamic behavior of a supply chain. As one of the major supply chain simulation methodologies, discrete-event simulation (DES) involves the modeling of a system in which the state variables change instantaneously at distinct points in time [47]. The method is commonly used to analyze complex processes that are challenging with closed-form analytical methods. Moreover, DES is widely used for supply chain management analysis, such as manufacturing process and logistics planning [40,51,72]. Simulations enable the design and evaluation of a supply chain prior to its implementation by performing a what-if analysis to ensure supply chain resiliency and sustainability [47,48,61].

A DES model is rarely run only once. Multiple simulation runs are usually required for various purposes. Because input parameters (e.g., the processing time of a product) are often random variables, multiple runs with different realizations of the random input variables are required to obtain statistically meaningful outputs. Furthermore, if a sensitivity analysis is applied on a simulation model to select input parameters that have the greatest effect on response variables, another layer of multiple runs is needed to vary input parameters, such as different distributions of processing times [50,53]. Optimization is another technique that can be combined with DES to define optimal input control variables (e.g., production capacity). Each optimization iteration requires multiple simulation runs for a set of control variables [27,42,44,60,70].

In summary, a large number of DES runs are often required for an analysis task. As the scale of supply chains grows large, for example, due to globalization and inter-enterprise collaboration [5,64], some simulation models may take hours to complete one run. Therefore, the time to perform analyses with thousands, sometimes hundreds of thousands, of DES runs for a complex supply chain can be prohibitively long when the standard Monte Carlo (MC) is used.

As an approximation of DES, the full simulation time can be divided into periods of given time buckets, Δt. Time-bucket based simulation does not model the occurrence of each event; instead, it counts the number of events happening in each time bucket, at the end of which the system state is updated using model equations. Therefore, in this

* Corresponding author.
E-mail addresses: chiangn@utrc.utc.com (N.-Y. Chiang), liny@utrc.utc.com (Y. Lin), longq@utrc.utc.com (Q. Long).

https://doi.org/10.1016/j.orp.2020.100144
Received 18 May 2019; Received in revised form 5 February 2020; Accepted 5 February 2020
Available online 10 February 2020

approach, events can be considered to occur instantaneously at the end of a period [48,68]. Note that our terminology "time bucket" is consistent with that in the supply chain literature (e.g., [68]) in which Δt can be equivalently denoted by "time interval", "time leap", and so on. The size of the time bucket can be defined either as a fixed value or in a time-dependent fashion. When the size of a time bucket is small enough that each bucket has at most one event, then the model is equivalent to DES. The advantage of the time-bucket method is that it is more scalable compared with DES when the size of the time bucket is relatively large. The disadvantage is that, due to the aggregation of multiple events, some interactions between events are lost; thus, the model is not as accurate as DES and is less commonly used.

The τ-leap method [13,16,36,54] is essentially a stochastic time-bucket method that has been widely used to accelerate the simulations of chemical reactions modeled by continuous-time Markovian processes. Rather than simulating every discrete event, the τ-leap method simulates the stochastic change of the system state at discrete time points using a constant propensity function to simulate the number of processes occurring during a time bucket. Although the simulation results become less accurate due to the time buckets, significant acceleration can be achieved under acceptable tolerance. Recently, the D-leap method has been proposed to accelerate the simulations of delayed chemical reactions [6] by introducing a queue of reactions to take into account the delays.

Our innovations are as follows. First, we extend the D-leap method to consider the distinctive dynamical features of the manufacturing supply chain and logistic networks in operational research. The resulting *L(logistic)-leap* method is able to consider production time, transportation time, limited capacity, inventory management, pull system and back orders. Secondly, we use the MLMC method based on time buckets to propagate the uncertainties in a supply chain, where most of the computational work is shifted from the expensive models (e.g., DES) to the inexpensive models defined by large time buckets. The proposed approach is able to match the model accuracy of DES while overcoming its scalability limitation with the help of the MLMC method. To the best of our knowledge, this is the first time that this type of leap method and the MLMC method have been used in supply chain management, which opens the door to more applications of large-scale problems in operations research and industrial practice.

This paper is organized as follows. Section 2 is a literature review on the DES in supply chain management and leap methods. Section 3 describes the accelerated approximation of DES using the time-bucket method and the detailed algorithms for the simulation of supply chain features. Section 4 introduces the L-leap method, which is a time-bucket method for simulating logistic systems driven by stochastic processes. Section 5 presents an MLMC method in which the samples are drawn from populations simulated using different sizes of time buckets. In Section 6, we show the accuracy and gain in computational speed using multiple examples. The first example models a push system, where the production does not depend on orders. The second example is a pull system with mixed orders of spare parts and final products. Transportation delays are also considered. The third example considers the uncertainty propagation of a push system under parametric uncertainties. The fourth example examines the uncertainty propagation of a pull system under both parametric uncertainties and those driven by stochastic processes. The quantities of interest are the final delivery time of a fixed number of orders and the number of deliveries over a specified time period. We show that the error of the predictive simulations with respect to (w.r.t.) the true solution provided by DES diminishes as we decrease the size of the time bucket. We achieve an acceleration factor of several orders of magnitude in computing the expected quantities of interest using the MLMC method based on the time buckets and L-leap, against the standard MC sampling.

2. Literature review

2.1. Discrete-event simulation in logistics and the supply chain

The DES approach is widely used in logistics and supply chain management as a tool to simulate the change in a system state over a time period of interest. For example, it has been used in supply chain network structures [1,9,12], inventory management [4,7,10,15,23–26], and supplier selection [21,22,41], and so on (see [66] for a detailed survey on the application of DES in the context of logistics and the supply chain). In DES, the system state variables change instantaneously at discrete time points when relevant events take place. While the definition of events is subject to the goal of the modeling, systematic approaches can be followed to design such a simulation [48]. The dominant type of DES is the next-event time-advance where the time clock always leaps to the most imminent time among the times of future events in an event list. The simulation complexity of DES is therefore proportional to the number of events in a real system during a simulated period, for a given number of processes. Distributed computation can be used to accelerate DES. Specifically, the implementations, such as the random number generator and the manipulation of the event list, can be parallelized. Moreover, a network can be decomposed into several sub-networks whose simulations can be parallelized. Many articles have been devoted to these topics, and detailed surveys can be found in [28,29,56,67]. Given an event list $\{event_i\}_{1\cdots N}$ and the corresponding time list $\{t_i^e\}_{1\cdots N}$, the event-driven DES can be summarized by the following pseudo code [30] of Algorithm 1.

2.2. Time-bucket method

In the time-bucket method, the system clock leaps forward to the next discrete time instance, and all the events occurring during the corresponding time-bucket are simulated. The time-bucket method can be viewed as a special case of next-event time-advanced DES [47,68]. The main advantage of the time-bucket method is that it can accelerate the simulation process if the predetermined time interval is large. However, because the system state only changes instantaneously at the end of each time bucket, rather than at the exact time when each event happens, it cannot provide an accurate timestamp for all events. For certain quantities of interest that need to be computed from the timestamp and system state, this method may introduce a substantial error if the predetermined time interval is too large. In general, the procedure and analysis of the time-bucket method have rarely been elaborated on in operations research literature for supply chain management. The general time-bucket method can be summarized in the following pseudo code of Algorithm 2.

2.3. τ-leap method for the approximation of DES in chemical and biochemical systems

The τ-leap method [36] is a widely used time-bucket method for the simulation of discrete chemical reactions. Rather than advancing the system clock to the next time instance when a reaction process takes place (e.g., Gillespie algorithm [35]), the τ-leap method predicts the number of reactions in a time interval using a random variable:

1: Input parameters: T, N, $\{event_i\}$, $\{t_i^e\}$
2: Initialize the timestamp $t \leftarrow 0$, $i \leftarrow 1$
3: **while** $t \le T$ AND $i \le N$ **do**
4: simulate $event_i$
5: if new event occurs, update $\{event_i\}$, $\{t_i^e\}$, and N
6: $t \leftarrow t_i^e$, $i \leftarrow i + 1$
7: **end while**

Algorithm 1. General Algorithm for Discrete Event Simulation.

1: Input parameters: T, Δt
2: Initialize the timestamp $t \leftarrow \min(\Delta t, T)$
3: **while** $t \leq T$ **do**
4: simulate all the events occurred during t and $t + \Delta t$
5: $t \leftarrow t + \Delta t$
6: update system vector
7: **end while**

Algorithm 2. General Algorithm for Time-Bucket Method.

$$\Delta c_p(t+\tau) = Poi(\tau \times r(t)), \tag{1}$$

where $\Delta c_p(t+\tau)$ represents the total number of processes, p, happening during $[t, t+\tau)$, and $Poi(\tau \times r(t))$ is a Poisson random variable with parameter $\tau \times r(t)$, where $r(t)$ is the rate function evaluated at time t. Based on the number of processes that occurred, we can update the system state variables (e.g., the numbers of products). Note that if $r(t)$ changes during the time period τ, the method introduces a time discretization error. However, the total simulation complexity is proportional to the number of time intervals, and the τ-leap method could be much faster than simulating every event for a given numerical tolerance. Efforts have been made to enhance the efficiency and accuracy of the original version of τ-leap, for instance, using efficient time interval selection [13], postleap checking [2], and a hybrid method [54]. In the τ-leap method, the reaction products are generated instantaneously without any delay after molecules collide. Its extension to delayed chemical reactions leads to the D-leap method [6].

2.4. D-leap method for simulating delayed chemical and biochemical systems

The D-leap method [6] is an extension of the τ-leap method in that it considers delayed chemical reactions. It counts the number of reactions occurring during a time interval using Equation(1) and the assumption is that the reactants are instantaneously consumed; hence, the system state is updated by the following:

$$x_i(t+\tau) = x_i(t) - \sum_p k_{pi}\Delta c_p(t+\tau) \quad for \quad i = 1, \ldots, n_s, \tag{2}$$

where x_i is the i^{th} system state variable, n_s is the number of system state variables, and k_{pi} is the consumption of x_i by a single event of the p^{th} reaction. The earliest possible production time is time t plus the given minimum delay of the p^{th} reaction, whereas the latest possible finishing time of Δc_p units of the p^{th} reaction is $t+\tau$ plus the given maximum delay of the p^{th} reaction. During any time interval that overlaps with the span between the earliest possible production time and the latest possible finishing time, the number of the accomplished p^{th} reactions, which is a fraction of $\Delta c_p(t+\tau)$, is defined by a binomially distributed random variable. Consequently, the system state is updated in a similar fashion as Equation (2). Nevertheless, the productions lead to positive changes in the number of the products w.r.t. a specific process. The products of a process can be the resources of a different process. The D-leap constitutes the basis for our *Logistic*-leap method in the logistic and supply chain contexts where the lead time of a process is usually non-negligible.

2.5. Monte Carlo method in supply chain management

The MC method is widely used to propagate uncertainties of random inputs to a typical quantity of interest in a supply chain [20,43,45,62,69]. Many variance reduction techniques [46,48] (e.g., antithetic variate and control variate) have been applied together with DES to increase the statistical efficiency of the uncertainty propagation. The MLMC method has emerged recently as a powerful sampling method to accelerate the computation of an expectation via drawing samples from a hierarchy of models [33,34], whereas control variate can be viewed as the simplest form of the MLMC method consisting of two levels [34]. In [3,55] the authors applied the MLMC method and the τ-leap method to the stochastic simulation of chemical reactions to achieve better scalability.

3. Time-bucket approximation of DES for supply chains

Supply chains transport materials from the suppliers to the places where they are consumed. The raw materials usually are consumed and transformed into intermediate products. We define the set $\mathbb{P}$ as all the parts, set $\mathbb{S} \subset \mathbb{P}$ as all the supplies of raw materials, and set $\mathbb{E} \subset \mathbb{P}$ as all the final products. For example, in the supply chain of the first numerical example (Fig. 3), we have eight parts, among which, three are raw materials, and one is the final product. Hence, $\mathbb{P} = \{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \mathcal{P}_4, \mathcal{P}_5, \mathcal{P}_6, \mathcal{P}_7, \mathcal{P}_8\}$, $\mathbb{S} = \{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3\}$, and $\mathbb{E} = \{\mathcal{P}_8\}$. The actual supply chain can be modeled as discrete mass flows with limited capacities (i.e., the production rate of each process is bounded from above). Specifically, a supply chain can be defined by a set of n processes, each of which can be described as follows:

$$\{\alpha_{ij}\hat{p}_{ij} | j = 1{:}\ \hat{n}_i\} \rightarrow \{\beta_{ik}\tilde{p}_{ik} | k = 1{:}\ \tilde{n}_i\} \quad i = 1, \ldots, n, \tag{3}$$

where, for each process i, $\hat{n}_i$ is the number of consumed parts, $\tilde{n}_i$ indicates the number of produced parts, $\hat{p}_{ij}$ denotes the j^{th} consumed part, and $\tilde{p}_{ik}$ represents the k^{th} produced part. We denote α_{ij} and β_{ik} as the integer weights corresponding to parts $\hat{p}_{ij}$ and $\tilde{p}_{ik}$, respectively. In other words, if process i happens once, it consumes α_{ij} units of part $\hat{p}_{ij}$ and produces β_{ik} units of part $\tilde{p}_{ik}$. Note that the symbols $\hat{p}_{ij}$ and $\tilde{p}_{ik}$ are "local" w.r.t. process i. A part may have different local symbols in different processes. For example, $\mathcal{P}_4$ is locally $\tilde{p}_{11}$ in the first process and $\hat{p}_{32}$ in the third process in the first example. By definition, $\mathbb{P}$ contains all the parts in the system; hence, we have $\mathbb{P} = \{\hat{p}_{ij}\} \cup \{\tilde{p}_{ik}\}$, for all i, j, and k. In addition, we have $\mathbb{S} = \{\hat{p}_{ij}\} \setminus \{\tilde{p}_{ik}\}$ and $\mathbb{E} = \{\tilde{p}_{ij}\} \setminus \{\hat{p}_{ik}\}$ for all i, j and k. We denote $\boldsymbol{x} \in \mathbb{Z}^{|\mathbb{P}|}$ as the state vector recording the number of parts, where $|\cdot|$ denotes the set cardinal. Note that the mapping $\{x\} \rightarrow \mathbb{P}$ is bijective, where $\{x\}$ is the set of the components of $\boldsymbol{x}$. Based on the definitions of $\hat{p}_{ij}$ and $\tilde{p}_{ik}$, we have $x_{\hat{p}_{ij}}$ as the number of the j^{th} parts consumed in the i^{th} process, and, similarly, $x_{\tilde{p}_{ik}}$ is the number of the k^{th} parts produced in the i^{th} process. For clarity, in the following text, we use $\hat{x}_{ij}$ and $\tilde{x}_{ik}$ to denote these quantities. At time t, the process occurs at a rate $\lambda_i(t)$, which is given by the following:

$$\lambda_i(t) = \begin{cases} \lambda_i^{max} & if \quad \min_j\{\hat{x}_{ij}(t) - \alpha_{ij}\lambda_i^{max}\Delta t\} \geq 0 \\ \min_j\left\{\frac{\lfloor \hat{x}_{ij}(t) \rfloor}{\alpha_{ij}}\right\} & otherwise \end{cases}, \tag{4}$$

where $\lambda_i^{\max}$ is the maximum production rate (capacity) associated with the i^{th} process, Δt is the size of the time bucket, and $\lfloor x \rfloor = \max\{m \in \mathbb{Z} | m \leq x\}$ is the floor function, which rounds x down to the nearest integer. The first expression in Equation (4) shows that the process can achieve its maximum rate if all the materials have sufficient inventory in this time bucket; Otherwise, the rate λ_i is reduced to the value that prevents negative values of the consumed materials during this time bucket. While Eq. (4) denotes a deterministic production rate, other alternatives are possible. For example, the consumption rate λ_i in Eq. (4) can be modeled by incorporating the expected arrivals of the consumed parts [19] (i.e., when one part, e.g., $\hat{p}_{ij}$, does not have sufficient inventory, its availability in the next time bucket may be estimated by checking the scheduled productions over this time bucket). If the number of scheduled productions plus the current inventory is larger than $\lambda_i^{\max}\Delta t$, the maximum capacity, $\lambda_i^{\max}$, can still be achieved. Otherwise, the consumption rate λ_i can be adjusted to match the summation of the expected arrival of $\hat{p}_{ij}$ and its current inventory. Nevertheless, we use Equation (4) in our approach

because it is more likely to prevent the negative inventory value of $\hat{p}_{ij}$. In the case in which a single part can be consumed by multiple processes, we assume this part is evenly consumed by all the following processes.

The time bucket simulation of a supply chain process can be split into two major phases. The first phase is material consumption, where each process consumes the necessary parts instantaneously according to its production rate - $\lambda_i(t)$. The second phase is delayed production, where due to the required processing time (lead time) in each process, we consider that all the productions require delays after materials have been instantaneously consumed. Our consumption-delayed-production framework follows the modeling procedures of the D-leap method for the delayed chemical reaction network simulation in [6]. More importantly, in the context of logistics, we enrich the D-leap method with several salient features of a supply chain: transportation, order-driven production (pull system), and priority production. We describe in details the time-bucket simulation of consumption-production in Sections 3.1 and 3.2.

3.1. Consumption

The consumption of parts during each time bucket is considered instantaneously at the beginning of every time bucket. In each time bucket Δt, the total number of the triggered i^{th} process is the following:

$$\Delta C_i(t) = \lambda_i(t)\Delta t\,. \tag{5}$$

The state vector is then updated by the following equation:

$$\hat{x}_{ij}(t) = \hat{x}_{ij}(t-\Delta t) - \alpha_{ij}\Delta C_i(t), \quad j = 1, \cdots, \hat{n}_i\,. \tag{6}$$

Note that it is possible to have a negative state variable $\hat{x}_{ij}(t)$ in (6). To prevent this, we enforce a correction step that offsets the computed number of triggered process by the amount related to the negative states. For the sake of conciseness, we omit the variable t and use ΔC_i instead of $\Delta C_i(t)$ in the remainder of this paper.

At each time point, we determine whether the executions of the ΔC_i processes should be completed or not, and estimate the number of completions, which is a fraction of ΔC_i. In the implementation, a queue structure is created to store the necessary information, i.e., the index of the delayed process d_{n_q}, where $n_q = 1, \ldots, N_q$, N_q is the number of process batches in the queue; the number of the delayed processes $Q^{delay}_{n_q}$; the earliest time of the production being completed $t^s_{n_q}$; and the time span between the earliest and latest times of the production being completed $t^{span}_{n_q}$.

The earliest production time and the total production period of the initiated processes can be computed as follows:

$$t^s_{n_q} = \; t + \hat{t}^{min}_{d_{n_q}}\,, \tag{7}$$

$$t^{span}_{n_q} = \; t + \Delta t + \hat{t}^{max}_{d_{n_q}} - t^s_{n_q} = \Delta t + \hat{t}^{max}_{d_{n_q}} - \hat{t}^{min}_{d_{n_q}}\,, \tag{8}$$

where $\hat{t}^{\min}_i$ and $\hat{t}^{\max}_i$ are the minimum and maximum lead times for each process i respectively. The definitions are schematically shown in Fig. 1.

We present the simulation algorithm of consumption for process i in Algorithm 3, which is a deterministic version of the consumption algorithm in [6].

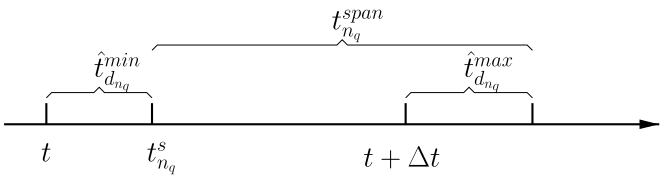

Fig. 1. Timeline of the processes started during Δt.

3.2. Delayed production

Productions are expected as long as $N_q \geq 1$. The simulation algorithm should determine whether any scheduled production is due to occur in the current time bucket (i.e., if $n_q \in \{1, \cdots, N_q\}$ exists that satisfies $t \leq t^s_{n_q} < t + \Delta t$). Assuming that the productions are uniformly distributed over time, the number of completed productions are proportional to the time fraction $t + \Delta t - t^s_{n_q}$ w.r.t. the total span $t^{span}_{n_q}$. Consequently, we update the associated components of the state vector $\{\hat{x}_{ik}\}$, $Q^{delay}_{n_q}$, $t^s_{n_q}$, and $t^{span}_{n_q}$. The details of the computations related to the delayed production are summarized in Algorithm 4, which is a deterministic version of the production algorithm in [6].

3.3. Push system

A push system, such as material requirement planning [59], controls the production flow moving from the supply end to the final retailer end according to the prediction of demands. Incorporating Algorithms 3 and 4, we present Algorithm 5, which simulates a push system of the supply chain. We may need to adjust the length of the last time bucket to ensure that the simulation stops at $t = T$ (lines 11–13 of Algorithm 5), where T is the end time of the simulation.

3.3.1. Inventory management

Inventory management is usually an important part of a push system. Safety stock is a popular and easy-to-implement remedy to mitigate disruptions in supply-chain operations [11,47,49,63], which can be caused by the temporal variations of product orders and the uncertainties in the supply. One strategy we can use to update the inventory is by adding the replenishment order quantity when the inventory is less than the safety stock, as follows:

$$x^b_p(t) = \begin{cases} S_p & if\ x_p(t) \leq x^s_p \\ 0 & otherwise \end{cases}, \tag{9}$$

where $p \in \mathbb{S}$ is a raw material, x^s_p is the safety stock, and S_p is a constant used as a safeguard for the stock of part p. Another possible way to place the replenishment order could be as follows:

$$x^b_p(t) = \begin{cases} x^s_p - x_p(t) + S_p & if\ x_p(t) \leq x^s_p \\ 0 & otherwise \end{cases},$$

which is more resilient toward uncertainties in the supply chain network. On the other hand, when we increase the amount of inventory, we expect increased storage costs. Finding a good balance between the safety stock x^s_p, safeguard S_p, order delay t^d_p, and costs remains challenging in practice. The optimal strategy for inventory management is problem specific, and extensive literature has been devoted to this topic [18,37,63,71].

Let $\bar{t}_p$ denote the time when the next supply of part p arrives. Given a constant $M > T$, our inventory management can be summarized as shown in Algorithm 6 for each time t when we update the system state.

3.4. Pull system

A pull system, such as the Toyota production system [58], also called just-in-time production or lean manufacturing, is a different policy design of manufacturing supply chain compared with a push design in that its production and inventory management are driven by incoming orders. In this section, we describe the time-bucket algorithms for order projection before we introduce the full time-bucket algorithm of the pull system. The inventory management simulation should remain the same as described in Section 3.3.1.

3.4.1. Projected order and pull system

Once a demand order is given, a supply chain system first determines whether sufficient inventory exists to meet the demand. If

Input parameters: $t, \Delta t, \vec{x}, \hat{n}_i, N_q, \lambda_i^{\max}, \{\alpha_{ij}\}, \hat{t}_i^{\min}, \hat{t}_i^{\max}$
Compute the total number of the triggered i^{th} processes using Equation
Update state vector $\{\hat{x}_{ij}\}$ using Equation
Check for negative values in $\{\hat{x}_{ij}\}$. If $\hat{x}_{iJ}$ is the smallest negative value in $\{\hat{x}_{ij}\}$, reduce the number of the triggered i^{th} processes by $-|\hat{x}_{iJ}|/\alpha_{ij}$ and repeat Step 3
Increase the queue length by one: $N_q \leftarrow N_q + 1$
Record the current process index in the queue structure: $d_{N_q} \leftarrow i$
Record the current consumption in the queue structure: $Q_{N_q}^{delay} \leftarrow \Delta C_i$
Compute the earliest production time $t_{N_q}^s$ and the production period $t_{N_q}^{span}$ using Equations and , respectively

Algorithm 3. Consumption.

Input parameters: t, Δt, N_q, $\{\tilde{n}_i\}$, $\vec{x}$, $\{\beta_{ik}\}$, $\{Q_{n_q}^{delay}\}$, $\{t_{n_q}^s\}$, $\{t_{n_q}^{span}\}$, $\{d_{n_q}\}$
for $n_q \in \{1, \ldots, N_q\}$ **do**
  Get the process index from queue structure: $i \leftarrow d_{n_q}$
  if $t_{n_q}^{span} > 0$ **AND** $t_{n_q}^s < t + \Delta t$ **then**
    Compute the number of finished processes in the current time bucket: $\Delta P_i \leftarrow Q_{n_q}^{delay} \min(\frac{t + \Delta t - t_{n_q}^s}{t_{n_q}^{span}}, 1)$
    Update state vector: $\tilde{x}_{ik}(t) \leftarrow \tilde{x}_{ik}(t - \Delta t) + \beta_{ik}\Delta P_i, \quad k = 1, \ldots, \tilde{n}_i$
    Update queue structure: $Q_{n_q}^{delay} \leftarrow Q_{n_q}^{delay} - \Delta P_i$
    Update queue structure: $t_{n_q}^s \leftarrow t + \Delta t$
    Update queue structure: $t_{n_q}^{span} \leftarrow \max(0, t_{n_q}^{span} - (t + \Delta t - t_{n_q}^s))$
  end if
end for

Algorithm 4. Production.

Input parameters: T, Δt, n, $\{\tilde{n}_i\}$, $\{\hat{n}_i\}$, $\{\alpha_{ij}\}$, $\{\beta_{ik}\}$, $\{\lambda_i^{\max}\}$, $\{\hat{t}_i^{\min}\}$, $\{\hat{t}_i^{\max}\}$ and $\{\hat{x}_{ij}(0)\}$
Initialize the queue length and the first time $t > 0$: $N_q \leftarrow 0$, $t \leftarrow \min(\Delta t, T)$
while $t \le T$ **do**
  $x(t) = x(t - \Delta t)$
  for all $i \in \{1, \ldots, n\}$ **do**
    Apply Algorithm 3 for consumption
  end for
  if $N_q \ge 1$ **then**
    Apply Algorithm 4 for productions
  end if
  if $t + \Delta t > T$ **then**
    $\Delta t \leftarrow T - t$
  end if
  $t \leftarrow t + \Delta t$
end while

Algorithm 5. Push System of a Supply Chain.

Input parameters: t, $\vec{x}$, $\{t_p^d\}$, $\{\bar{t}_p\}$, M, $\{x_p^s\}$, $\{S_p\}$
for all $p \in \mathbb{S}$ **do**
  Compute the back order quantity x_p^b using
  if $t \ge \bar{t}_p$ **then**
    Back order arrived. Add it into the state vector: $x_p \leftarrow x_p + x_p^b$
    Reset the next arrival time: $\bar{t}_p \leftarrow M$
  else if $t < \bar{t}_p$ **AND** $\bar{t}_p = M$ **AND** $x_p^b > 0$ **then**
    Compute the next back order arrival time: $\bar{t}_p \leftarrow t + t_p^d$
  end if
end for

Algorithm 6. Inventory Management.

there is insufficient inventory to fulfill the demand, the supply chain needs to start the production to fill the gap. Hence, we need to backtrack to determine whether the existing inventories of all the intermediate parts can satisfy their own demands.

To guarantee that all the demands are satisfied, the projected accumulated demand g_p, which includes the number of parts that are consumed in the intermediate processes, should be calculated using the following recursive function:

$$g_p(t) = \begin{cases} \hat{g}_p(t) & \text{if} \quad p \in \mathbb{E} \\ \sum_{\{(i,j)|\hat{p}_{ij}=p\}} \alpha_{ij} \max_k \left(\lceil \frac{g_{\tilde{p}ik}}{\beta_{ik}} \rceil \right) + \hat{g}_p(t) & \text{otherwise} \end{cases}, \tag{10}$$

where $\hat{g}_p(t) = \sum_{\tau \le t} g_p^*(\tau)$ is the total order of part $p \in \mathbb{P}$ cumulative in time up to t, where $\{\tau\}$ are discrete time points in the simulation, $g_p^*(\tau)$ is the incoming order of part $p \in \mathbb{P}$ at time τ, and $\lceil x \rceil = \min\{m \in \mathbb{Z} | m \ge x\}$ is the ceiling function, which rounds x up to the nearest integer. The second expression of Equation (10) consists of the direct order of part p and the demand associated with its "offspring" parts: $\sum_{\{(i,j)|\hat{p}_{ij}=p)\}} \alpha_{ij} \max_k \left(\lceil \frac{g_{\tilde{p}ik}}{\beta_{ik}} \rceil \right)$.

This recursive projection can be visualized by a process starting from the final product. For example, consider a small supply chain that involves four parts as shown in Fig. 2. Assume we have some spare part orders at time t for parts B and D, and each order requires 100 units. Using backward recursion (10), we can obtain the projected demands for parts A, B, C, and D as 200, 200, 100, and 100, respectively.

The projected value g_p indicates the necessary quantity of part p that must be produced to satisfy the given orders. Quantity $g_p - \hat{g}_p$ represents the least amount of part p that should be consumed in the related processes. The estimated consumption according to Eq. (5) may be larger than $g_p - \hat{g}_p$ during a given Δt. In this situation, we introduce a variable *flag* to control the consumption. The value of *flag* is determined by comparing the accumulated consumption $c_p(t) = \sum_{\tau \le t} \sum_{\{i|\hat{p}_{ij}=p\}} \alpha_{ij} \Delta C_i(\tau)$ with $g_p - \hat{g}_p$. If $c_p(t) > = g_p - \hat{g}_p$, then part p has already been consumed sufficiently, and no more consumption should happen. The projected order and pull strategy is summarized in Algorithm 7. Finally, Algorithm 8 lays out the simulation iterations of a pull system.

Remark 1. The simulation using the proposed algorithms (Algorithms 5 and 8) converges to the results from the DES, when the time interval Δt is small enough such that each individual event is resolved in the simulation. We numerically show this in the first example.

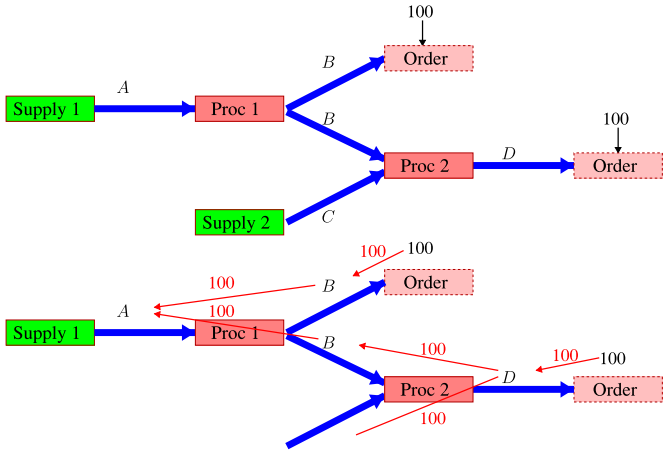

Fig. 2. The flowchart on top shows two orders on parts B and D, and each order requires 100 units. The bottom flowchart shows the projected demand of each part.

3.5. Hybrid system

A hybrid system is a combination of push and pull strategies [31,32,38,39]. In a hybrid system, some of the production stages are organized by push strategies due to a low level of uncertainty of the demand from their following stages; the production at the other stages, such as the final assembly, is organized by a pull strategy due to the high level of demand uncertainty. The corresponding time-bucket implementation would combine the push and pull strategies described in the previous sections on a system level.

4. Stochastic time-bucket method: L-leap

In the previous sections, we presented the deterministic time-bucket approximation of DES, where the number of processes occurring during a fixed time interval is a deterministic value (i.e., $\Delta C_i = \lambda_i(t)\Delta t$, $i = 1, \cdots, n$). By introducing randomnesses into the simulation, it allows us to better understand the potential risk in the supply chain system. Similar to D-leap [6], we treat both the number of triggered processes, ΔC_i, and the number of accomplished processes, ΔP_i, as random variables. Note that our framework can extend easily to model the uncertainties from other sources, such as the demand and supply.

We use the Poisson distribution to model the number of processes occurring in Δt with parameter $\lambda_i \Delta t$ [6,48]:

$$\Delta C_i \sim Poi(\lambda_i \Delta t), \quad i = 1, \cdots, n, \tag{11}$$

and the binomial distribution to model the number of productions [6,48] in $t + \Delta t - t_{n_q}^s$ given the fixed number of production, $Q_{n_q}^{delay}$, during $t_{n_q}^{span}$:

$$\Delta P_{d_{n_q}} \sim B(Q_{n_q}^{delay}, \min(\frac{t + \Delta t - t_{n_q}^s}{t_{n_q}^{span}}, 1)) \quad n_q = 1, \cdots, N_q . \tag{12}$$

Note that binomial sampling leads to uniformly distributed production over the production span in the continuous time limit as $\Delta t \to 0$.

In addition, Algorithms 3 and 4 can be easily extended to their stochastic version using (11) and (12). The stochastic consumption and production can be embedded in the simulation flow of Algorithm 8, which leads to a new stochastic simulation strategy. We call it the L (logistic)-leap method, where we use a constant average production rate, Boolean values associated with the inventory policies, and the order projections at time t to predict the productions during t and $t + \Delta t$. There is a finite probability at every time step that the computed number of parts becomes negative because the number of processes ΔC_i generated in (11) can be arbitrarily large. To prevent this, we enforce a correction step that effectively constrains the Poisson distribution to a feasible domain.

Compared with the exact simulation of the DES, the approximation used here has the flexibility of accelerating the computation under prescribed numerical tolerance. Indeed, we show that uncertainty propagation in a supply chain can be dramatically accelerated without sacrificing any accuracy if we use the time-bucket simulation in a coordinated way. The L-leap method we are using has a piece-wise constant rate function, and its stability can be proved using the approach described in [14].

5. Uncertainty propagation using time bucket simulation and the MLMC method

In this section, we describe the problem of colorredforward uncertainty propagation, the MC discretization of an expectation, and the MLMC approach to compute the expectation. The MLMC method was combined with τ-leap for uncertainty quantification in the context of stochastic chemical reactions in [3,55].

Forward uncertainty propagation is concerned with the estimation of the expected value of a quantity of interest(q), such as the delivery

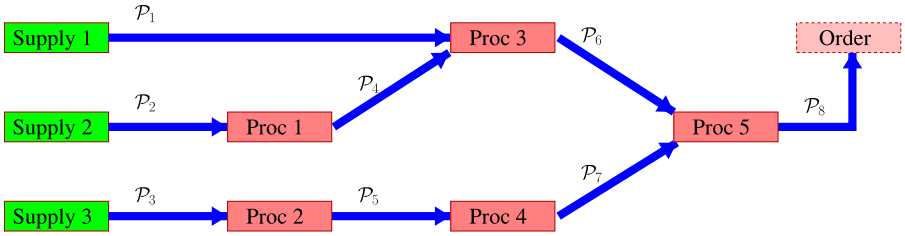

Fig. 3. A manufacturing system with five processes and eight parts in Section 6.1.

time of the final products. The standard MC estimator is given by the following

$$\boldsymbol{E}_{\theta,\omega}[q(\theta,\omega)] = \frac{1}{N_s}\sum_{k=1}^{N_s} q(\theta_k,\omega_k) + O_P\left(\frac{1}{\sqrt{N_s}}\right), \tag{13}$$

where $\boldsymbol{\theta}$ is the vector of random parameters, ω is the noise that perturbs the system state variables dynamically, and N_s is the number of samples. The notation of a sequence of random variables $Y_{N_s} = O_P(d_{N_s})$ indexed by N_s means that, for any $\epsilon > 0$, a finite K and a finite N_0 exist, such that, for any $N_s > N_0$, the probability $Pr(Y_{N_s} > Kd_{N_s})$ is smaller than ϵ. Assigning a tolerance ϵ_s and a confidence level α on the statistical error leads to the following

$$Pr\left(\left|\frac{1}{N_s}\sum_{k=1}^{N_s} q(\theta_k,\omega_k) - \boldsymbol{E}_{\theta,\omega}[q(\theta,\omega)]\right| < \epsilon_s\right) = \alpha\,. \tag{14}$$

Considering the central limit theorem (CLT) (i.e., $\frac{1}{N_s}\sum_{k=1}^{N_s} q(\theta_k,\omega_k) - \boldsymbol{E}_{\theta,\omega}[q(\theta,\omega)] \sim \mathcal{N}(0,\frac{V}{N_s})$ as $N_s \to \infty$), we can equivalently express Eq. (14) using the distribution function of a standard normal:

$$\Phi\left(\frac{\sqrt{N_s}\epsilon_s}{\sqrt{V}}\right) = \frac{1+\alpha}{2}\,.$$

Consequently, we obtain the expected number of samples to control the statistical error in probability:

$$N_s = V c_\alpha^2 \epsilon_s^{-2}\,, \tag{15}$$

where V is the variance of the quantity of interest, $c_\alpha = \Phi^{-1}(\frac{1+\alpha}{2})$ is the inverse distribution function of the standard normal distribution, and ϵ_s is the tolerance on the absolute error of the MC estimator. Then, the total computational cost of a standard MC sampler is as follows:

$$C_{mc} = C_m V c_\alpha^2 \epsilon_s^{-2}\,, \tag{16}$$

where C_m is the average cost of a single DES.

The MLMC method is optimized in the sense that the total computational cost is minimized for a given tolerance of the numerical error. In the hierarchy of models, high-level models are more accurate and computationally more expensive than low-level models. Provided that the expectation and variance of the difference between the approximate and true solutions diminish at certain rates as the level increases, we can construct an MLMC sampler that can be several orders more efficient than the standard MC method. Note that the standard MC method would put all of its samples on the highest level to control the bias of the estimator. Let q_l denote the corresponding level l approximation of the quantity of interest q. Assume that the numerical discretization error is bounded uniformly in the probability space as follows:

$$\boldsymbol{E}(q - q_l) = O(\Delta t_l^a)\,, \tag{17}$$

where Δt_l is the size of the time bucket on level l, $a \in \mathbb{R}^+$ is the convergence rate of the numerical discretization, and the notation $Y_{\Delta t} = O(d_{\Delta t})$ indexed by Δt is the deterministic version of $Y_{\Delta t} = O_P(d_{\Delta t})$, which means that a finite K and a finite Δt_0 exist, such that, for any $\Delta t < \Delta t_0$, $Y_{\Delta t} \le K d_{\Delta t}$.

The expectation in Equation (13) can be rewritten as a telescopic sum as follows:

$$\boldsymbol{E}(q) = \sum_{l=0}^{L} \boldsymbol{E}(q_l - q_{l-1}) + O(\Delta t_L^a)\,, \quad \text{with} \quad q_{-1} = 0\,. \tag{18}$$

Furthermore, we can write the first term on the right hand side (r.h.s.) of Eq. (18) as a summation of sample averages, and Eq. (18) becomes the following:

$$\boldsymbol{E}(q) = \hat{q} + \sum_{l=0}^{L} O_P\left(\frac{1}{\sqrt{N_l}}\right) + O(\Delta t_L^a)\,, \quad \text{with} \quad q_{-1} = 0\,, \tag{19}$$

where

$$\hat{q} = \sum_{l=0}^{L}\frac{1}{N_l}\sum_{k=1}^{N_l}(q_l^k - q_{l-1}^k)\,, \tag{20}$$

is the MLMC estimator of q, $\sum_{l=0}^{L} O_P\left(\frac{1}{\sqrt{N_l}}\right)$ is the statistical error, and $O(\Delta t_L^a)$ is the numerical bias. A heuristic argument on the computational advantage of using this estimator is the following: the variance of $q_l - q_{l-1}$ becomes very small as l increases; hence, few high-level samples are drawn while most of the samples are shifted to the lower levels where the computations are fast.

Next, we optimize the computational cost of the MLMC estimator for given tolerances on the bias and statistical error, which are written as follows:

Input parameters: n, $\{\hat{g}_p\}$, $\{c_p\}$, $\{\alpha_{ij}\}$, $\{\beta_{ik}\}$
for all $i \in \{1,\ldots,n\}$ **do**
  $flag_i \leftarrow 0$
end for
for all $p \in \mathbb{P}$ **do**
  compute the projected accumulated demand g_p using
  if $c_p < g_p - \hat{g}_p$ **then**
    for all $i \in \{i \mid \exists \hat{p}_{ij} = p\}$ **do**
      the process that produces p still needs to be continued by setting $flag_i \leftarrow 1$
    end for
  end if
end for

Algorithm 7. Projected Order and Pull Strategy.

Input parameters: T, Δt, n, $\{\tilde{n}_i\}$, $\{\hat{n}_i\}$, $\{\alpha_{ij}\}$, $\{\beta_{ik}\}$, $\{\lambda_i^{\max}\}$, $\{\hat{t}_i^{\min}\}$, $\{\hat{t}_i^{\max}\}$, $\{\hat{x}_{ij}(0)\}$, $\{t_p^d\}$, $\{\bar{t}_p\}$, M, $\{x_p^s\}$, $\{S_p\}$, $\{\hat{g}_p\}$
Initialize the queue length $N_q \leftarrow 0$; timestamp $t \leftarrow \min(\Delta t, T)$; next back order arrival time $\{\bar{t}_p\} \leftarrow M$; accumulated consumption $\{c_p\} \leftarrow 0$
while $t \le T$ **do**
  $x(t) = x(t - \Delta t)$
  Apply Algorithm 6 to compute back order
  Apply Algorithm 7 to compute the projected order quantities
  for all $i \in \{1, \dots, n\}$ **do**
    if $flag_i > 0$ **then**
      Apply Algorithm 3 for consumption
      for all $j \in \{1, \dots, \hat{n}_i\}$ **do**
        Update the accumulated consumption: $c_{\hat{p}_{ij}} \leftarrow c_{\hat{p}_{ij}} + \alpha_{ij}\Delta C_i$
      end for
    end if
  end for
  if $N_q \ge 1$ **then**
    Apply Algorithm 4 for productions
  end if
  if $t + \Delta t \ge T$ **then**
    $\Delta t \leftarrow T - t$
  end if
  $t \leftarrow t + \Delta t$
end while

Algorithm 8. Time-Bucket Simulation of the Supply Chain-Pull System.

$$\boldsymbol{E}(q - q_L) = \epsilon_b\,, \tag{21}$$

$$Pr(|\hat{q} - \boldsymbol{E}(q_L)| < \epsilon_s) = \alpha\,, \tag{22}$$

where ϵ_b is the tolerance for the bias and ϵ_s is the tolerance for the statistical error. Note that we can use the CLT to convert Eq. (22) to the following variance constraint:

$$\boldsymbol{Var}(\hat{q}) = \frac{\epsilon_s^2}{c_\alpha^2}\,. \tag{23}$$

The maximum level can be obtained from Eqs. (17) and (21):

$$L = \frac{1}{a} log_2(\epsilon_b)\,,$$

assuming that $2^{-l} = \Delta t_l$.

The optimal number of samples on each level can be obtained by minimizing the total cost under the constraint of Eq. (23) on the variance of the estimator:

$$\{N_l^{opt},\ l = 0, ..., L\} = \underset{\{N_l,\, l=0,\dots,L\}}{\arg\min}\left[\sum_{l=0}^{L} C_l N_l + \lambda\left(\sum_{l=0}^{L}\frac{V_l}{N_l} - \frac{\epsilon_s^2}{c_\alpha^2}\right)\right],$$

where C_l is the average computational cost of $q_l - q_{l-1}$, V_l is the variance of the random variable $q_l - q_{l-1}$ and λ is a Lagrangian multiplier (by an abuse of notation).

Solving the above minimization problem leads to the following

$$N_l^{opt} = \sqrt{\frac{V_l}{C_l}}\frac{\sum_{l=0}^{L}\sqrt{C_l V_l}}{\bar{\epsilon}_s^2} \quad \text{with} \quad \bar{\epsilon}_s^2 = \frac{\epsilon_s^2}{c_\alpha^2}\,.$$

Consequently, the optimal total computational cost of the multilevel estimator is as follows:

$$\sum_{l=0}^{L} C_l N_l^{opt} = \sum_{l=0}^{L} C_l \sqrt{\frac{V_l}{C_l}}\frac{\sum_{l=0}^{L}\sqrt{C_l V_l}}{\bar{\epsilon}_s^2} = \left(\sum_{l=0}^{L}\sqrt{C_l V_l}\right)^2 \bar{\epsilon}_s^{-2}\,.$$

It is common that the variance V_l and cost C_l have the asymptotic bounds: $V_l = O(\Delta t_l^b)$ and $C_l = O(\Delta t_l^{-g})$, where b and g are the rates that describe the algebraic decrease/increase of the variances and computational costs, respectively. In the cases where $C_0V_0 > > C_1V_1 > > \cdots > > C_LV_L$, the total cost is represented by $C_0V_0\bar{\epsilon}_s^{-2}$. In the cases where $C_LV_L > > C_{L-1}V_{L-1} > > \cdots > > C_0V_0$, the total cost is represented by $C_LV_L\bar{\epsilon}_s^{-2} = C_0V_0\epsilon_b^{-\frac{g-b}{a}}\bar{\epsilon}_s^{-2}$. In the cases where $C_LV_L = C_{L-1}V_{L-1} = \cdots = C_0V_0$, the total cost is $L^2C_0V_0\bar{\epsilon}_s^{-2} = C_0V_0(log_2\epsilon_b)^2\bar{\epsilon}_s^{-2}/a^2$. Note that, in the literature, it is common to impose a total tolerance ϵ^2 on the mean square error of the MLMC estimator and to split the error budget into two parts - $\theta\epsilon^2$ and $(1-\theta)\epsilon^2$ where $(0 < \theta < 1)$ on the bias and variance [33,34,52]. We give an explicit confidence level to the statistical error control in this study, which is consistent with the literature [8,17,55].

Remark 2. In the case where $C_L \approx C_m$, the complexity of a standard MC sampler is $O\left(\bar{\epsilon}_s^{-2}\epsilon_b^{-\frac{g}{a}}\right)$, where $\bar{\epsilon}_s^{-2}$ is proportional to the number of samples of a standard MC method and $\epsilon_b^{-\frac{g}{a}}$ is proportional to C_L, the computational cost of each sample on the highest level L. Therefore, the computational complexity of MC (16) will always be asymptotically higher than those that can be achieved by the MLMC method, namely $O(\bar{\epsilon}_s^{-2})$, $O\left(\epsilon_b^{-\frac{g-b}{a}}\bar{\epsilon}_s^{-2}\right)$, and $O((log_2\epsilon_b)^2\bar{\epsilon}_s^{-2})$.

Remark 3. The terms q_l^k and q_{l-1}^k in Eq. (20) should always be computed using the same realization of the random parameters as their inputs to ensure the correlation between q_l and q_{l-1}. In the cases in which the randomness is driven by stochastic processes, we adopt the coupling scheme proposed in Algorithm 2 of [3]. The key idea of this algorithm is to use the additivity property of Poisson processes to tightly correlate two processes on different levels. For the sake of completeness, we describe the coupling procedure as follows. The processes associated with q_l^k and q_{l-1}^k at time steps t, $t + \Delta t_l$, and $t + \Delta t_{l-1}$ are simulated as in Algorithm 9. In this way, two processes on two consecutive levels can be coupled, and the variance of $q_l - q_{l-1}$ can be effectively reduced. This approach is used in our final example. We

for $j = 0, 1$ **do**
$A_1 = \min(\lambda_i(\hat{x}_{ij}^{l}(t + j \times \Delta t_l)), \lambda_i(\hat{x}_{ij}^{l-1}(t_n)))$
$A_2 = \lambda_i(\hat{x}_{ij}^{l}(t + j \times \Delta t_l)) - A_1$
$A_3 = \lambda_i(\hat{x}_{ij}^{l-1}(t)) - A_1$
$\Delta C_1 = Poi(A_1 \times \Delta t_l)$
$\Delta C_2 = Poi(A_2 \times \Delta t_l)$
$\Delta C_3 = Poi(A_3 \times \Delta t_l)$
$\hat{x}_{ij}^{l}(t + (j+1) \times \Delta t_l) = \hat{x}_{ij}(t + j \times \Delta t_l) - \alpha_{ij} \times (\Delta C_1 + \Delta C_2)$
$\hat{x}_{ij}^{l-1}(t + (j+1) \times \Delta t_l) = \hat{x}_{ij}(t + j \times \Delta t_l) - \alpha_{ij} \times (\Delta C_1 + \Delta C_3)$
end for
$t = t + \Delta t_{l-1}$

Algorithm 9. Algorithm to couple the consumptions on levels l and $l-1$.

can likely further reduce the variance of $q_l - q_{l-1}$ by coupling the binomial random variables in Eq. (12), which are associated with two consecutive levels. One option is to first convert the binomial random distributions to their approximate Poisson random distributions, whose average number of events per interval equals the mean of the corresponding binomial random distributions [57], and then use the additivity property of Poisson processes to perform the coupling. Nevertheless, we achieved accelerated results, although we did not couple the binomial random variables in the last example because the coupling of the Poisson processes on two consecutive levels provided an effective reduction of variances.

6. Experimental validation

In this section, we present four examples of numerical experiments with increasing complexities. The first example is a manufacturing material flow simulated using deterministic and stochastic time-bucket methods. The second example is a pull system considering inventory replenishment, priority delivery, and transportation delays simulated using time-bucket methods. We carried out uncertainty propagation using the MLMC method in the third and fourth examples. We used MATLAB to implement the time-bucket algorithm and built our code of the MLMC method on the original version from https://people.maths.ox.ac.uk/gilesm/mlmc/.

6.1. time-bucket approximations of a simple push supply chain network

We considered a supply chain system for manufacturing industry, which is schematically shown in Fig. 3. It involves five processes and eight parts, and we show the consumption-production relationships in Eqs. (24)–(28). The parts on the left hand side of the equations are instantaneously consumed when the processes are started, while the parts on the r.h.s. are produced after certain periods of delays, characterized by the production time/lead time of each process. The production rate, which describes the capacity of a process, is the number of parts that are processed in a time unit (e.g., one day).

$$\mathcal{P}_2 \to \mathcal{P}_4 \tag{24}$$

$$\mathcal{P}_3 \to \mathcal{P}_5 \tag{25}$$

$$\mathcal{P}_1 + \mathcal{P}_4 \to \mathcal{P}_6 \tag{26}$$

$$\mathcal{P}_5 \to \mathcal{P}_7 \tag{27}$$

$$\mathcal{P}_6 + \mathcal{P}_7 \to \mathcal{P}_8 \tag{28}$$

A push system starts the production process according to the demand prediction. We assume the following initial conditions: $x_1(t=0) = 1000$, $x_2(t=0) = 500$, and $x_3(t=0) = 1000$, which describe the initial inventory levels of $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$. For all $i \notin \mathbb{S}$ (i.e., the intermediate and final products), we let $x_i(t=0) = 0$. First, we assume the deterministic production rates of $\lambda_1 = 8$, $\lambda_2 = 8$, $\lambda_3 = 4$, $\lambda_4 = 8$, and $\lambda_5 = 2$. We also assume the processing time is deterministic (i.e., $\hat{t}_i^{\max} = \hat{t}_i^{\min}$, $i = 1, \cdots, 5$, which are $\hat{t}_1^{\min} = 1$, $\hat{t}_2^{\min} = 1$, $\hat{t}_3^{\min} = 10$, $\hat{t}_4^{\min} = 1$, and $\hat{t}_5^{\min} = 10$).

Fig. 4 shows the time histories of the state vector, which represents the number of each part in the system at any given time, simulated under two different values of the time bucket. The time-bucket approximation can capture the main dynamic features of the system, even when a coarse time bucket size, $\Delta t = 16$ days, is used. The monotonic decrease of $\mathcal{P}_1$ stops at 500 parts due to the initial inventory level of $\mathcal{P}_2$. In addition, x_8 monotonically increases after an initial period of waiting, which is attributed to the production delays. The dynamics of the intermediate parts, $\mathcal{P}_4$, $\mathcal{P}_5$, $\mathcal{P}_6$, and $\mathcal{P}_7$, are primarily determined by their consumption and production rates.

Fig. 5 illustrates that the time-bucket method converges to the "ground truth" computed using DES when Δt reduces from 32 days to 4 days. The error is smaller than 2% when the time bucket is smaller than 4 days.

The absolute error of the 200-day production decreases linearly when the time-bucket size decreases as shown on the left in Fig. 6. The CPU time of the time-bucket approximation increases linearly as the number of time buckets increases during the simulation time (the CPU time is an average value over 100 repetitive runs).

Next, we use the L-leap method to approximately simulate the stochastic system where the state vector is dynamically driven by a Poisson process. Fig. 7 visualizes 1000 trajectories using identical initial data. In addition, the average trajectories shift from left to right when we reduce Δt; for example, the 500th $\mathcal{P}_8$ is produced in around 300 days when $\Delta t = 16$ days, while it is produced in around 270 days with $\Delta t = 2$ days. This is due to the artificially delayed availability of its previous parts when the time bucket is coarse.

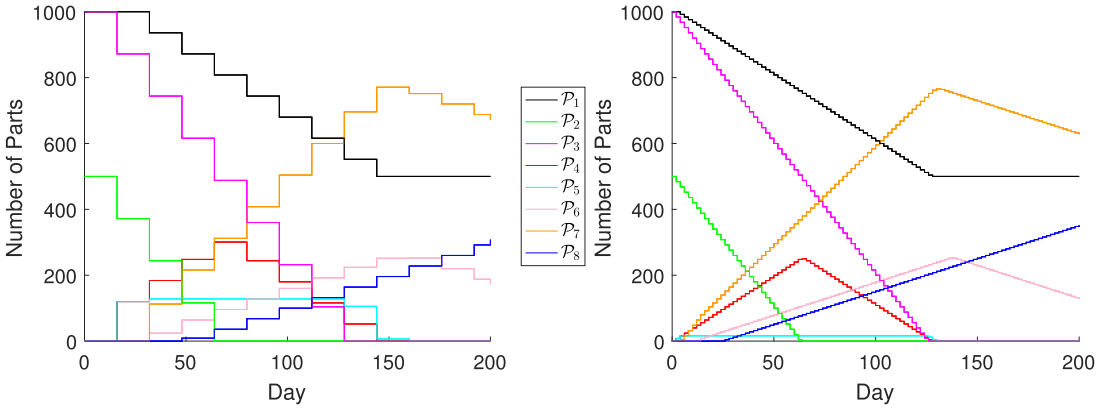

Fig. 4. The time history of the state vector in the push system in Section 6.1. From left to right, figures present the case for $\Delta t = 16$ days and $\Delta t = 2$ days, respectively.

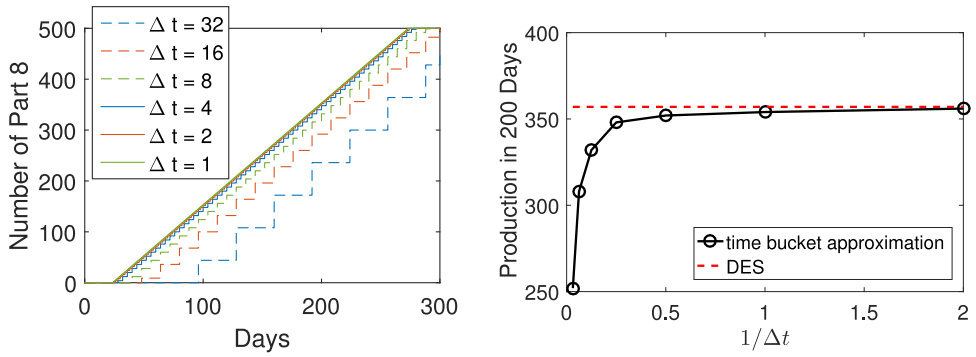

Fig. 5. Push system in Section 6.1. Left figure is the simulated evolution of the number of final products ($\mathcal{P}_8$); right figure is the convergence of the number of products produced in 200 days w.r.t. the reciprocal of the size of time bucket. The reference value is 357.

6.2. time-bucket approximations of a complex pull system

This example is a pull system dealing with mixed orders of spare parts and final products while considering transportation time. The system receives spare part orders of $\mathcal{P}_4$, $\mathcal{P}_5$, $\mathcal{P}_6$, and $\mathcal{P}_7$ every 50 days. Meanwhile, the following inventory policy is adopted to refill $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$: when the quantity of a part in inventory falls below 200, replenishment orders of 200, 250, and 300 are placed for $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$, respectively. The delivery delays are 15, 20, and 30 days for $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$, respectively. Moreover, each of the parts has an initial inventory of 500. In addition to the spare part orders, we place three orders of final products on the first day, the 100th day, and the 200th day, and each order consists of 100 final products $\mathcal{P}_8$. Upon the receipt of final product orders, priority is given to the final product production over spare part orders.

Additionally, we include the transportation time between consecutive processes. The transportation of products is modeled as additional processes characterized by transportation rates and transportation delay time (similar to the production rates and processing time of a production process). The new system is shown in Fig. 8, where the first five processes are the original processes, and the second set of five processes are the transportation processes. For Processes 6 to 10, we use transportation rates $\lambda_i = 8$, $\forall\, i \in \{6, \cdots, 10\}$ and the constant transportation delay time $\hat{t}_i^{\min} = \hat{t}_i^{\max}$, at 10, 10, 10, 50, and 10 days.

Fig. 9 shows the simulated numbers of parts in the system as they change over time using two different lengths of time buckets (i.e., $\Delta t = 16$ and 2 days). The start of the delivery of the final product $\mathcal{P}_{13}$ has been shifted to a later date compared to that of the case without transportation. Due to the creation of the new processes, we can simulate the number of goods in the buffers right after their production, during the transportation, and in the buffers before their instantaneous consumption in the following process.

Additionally, we simulate a stochastic pull system with mixed orders and transportation using the L-leap method. We present the average number and its 95% confidence interval in 700 days for part $\mathcal{P}_3$ in Fig. 10. Very large uncertainties exist at the points where the inventory can be refilled.

6.3. uncertainty propagation using the MLMC method - push system

We use the MLMC method to compute the expected number of $\mathcal{P}_8$ delivered in 300 days in the previous push system. We consider 13 random parameters (i.e., $\lambda_1 - \lambda_5$ are the production rates of the processes $1 - 5$, $x_1(t=0)$, $x_2(t=0)$, and $x_3(t=0)$ are the initial inventories of $\mathcal{P}_1 - \mathcal{P}_3$, and $\hat{t}_i$, $i = 1, \cdots, 5$ are the processing times of processes $1 - 5$). The parameters are independently uniformly distributed as follows:

$$
\begin{aligned}
&\lambda_1 \sim \mathcal{U}(8, 12), \quad \lambda_2 \sim \mathcal{U}(8, 12), \quad \lambda_3 \sim \mathcal{U}(4, 6),\\
&\lambda_4 \sim \mathcal{U}(8, 12), \quad \lambda_5 \sim \mathcal{U}(1, 3),\\
&x_1(t=0) \sim \mathcal{U}(800, 1200), \quad x_2(t=0) \sim \mathcal{U}(300, 700),\\
&x_3(t=0) \sim \mathcal{U}(800, 1200),\\
&\hat{t}_1 \sim \mathcal{U}(1, 2), \quad \hat{t}_2 \sim \mathcal{U}(1, 2), \quad \hat{t}_3 \sim \mathcal{U}(10, 20), \quad \hat{t}_4 \sim \mathcal{U}(1, 2),\\
&\hat{t}_5 \sim \mathcal{U}(10, 50).
\end{aligned}
$$

We evenly split the total tolerance between the bias and statistical error (i.e., $\epsilon_b^2 = 0.5TOL^2, \bar{\epsilon}_s^2 = 0.5TOL^2$). The estimated values of a, b, and g are 1.5, 2, and 1, respectively. They are empirically estimated as the slopes of the regression curves that fit the logarithm of the data, namely the sample averages, variances, and computational costs of $q_l - q_{l-1}$ against the first five levels. To understand the errors of the estimated a, b, and g, we repeat the regression process 100 times, during which 10,000 independent samples of $q_l - q_{l-1}$ are used on every level. The sample averages and variances of $q_l - q_{l-1}$ are plotted in Fig. 11 (a) and (b), respectively. The regression results are $a = 1.31 \pm 0.06$, $b = 2.03 \pm 0.09$, and $g = 0.97 \pm 0.15$ with a 99.7% confidence interval.

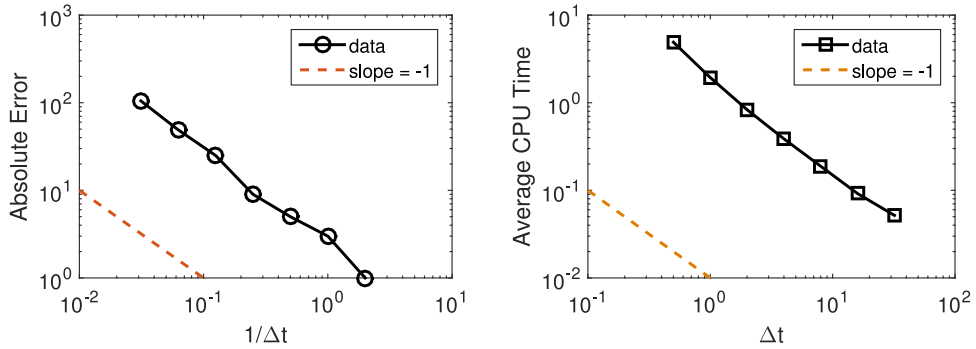

Fig. 6. Push system Section 6.1. Left figure is the absolute error of the 200 days production w.r.t. the reciprocal of the size of the time bucket. Right figure is the CPU time averaged over 100 repetitive runs of the simulation up to 200 days, w.r.t. the size of the time bucket.

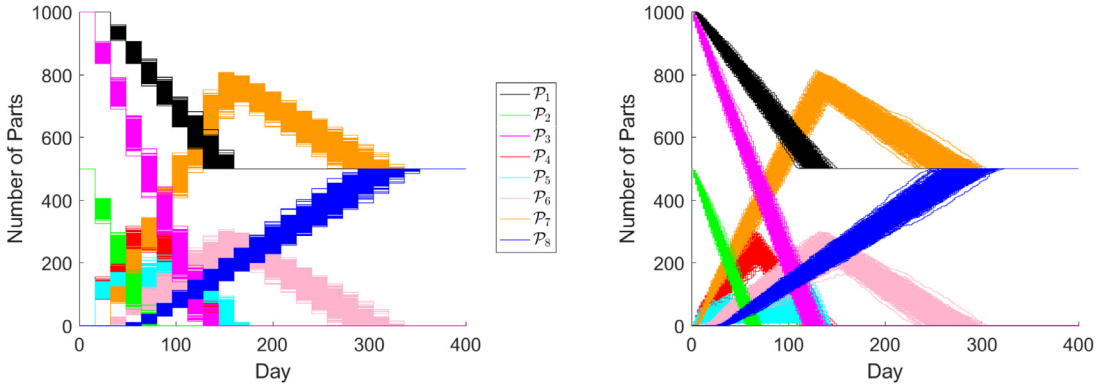

Fig. 7. The time history of the state vector in a stochastic push system in Section 6.1. From left to right, figures present the case for $\Delta t = 16$ days and $\Delta t = 2$ days, respectively.

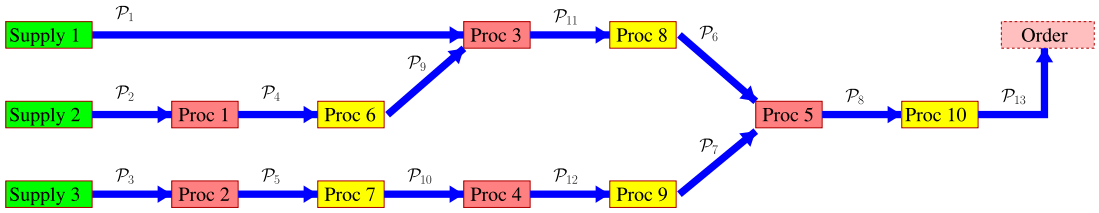

Fig. 8. A modified manufacturing system which includes transportation in Section 6.2.

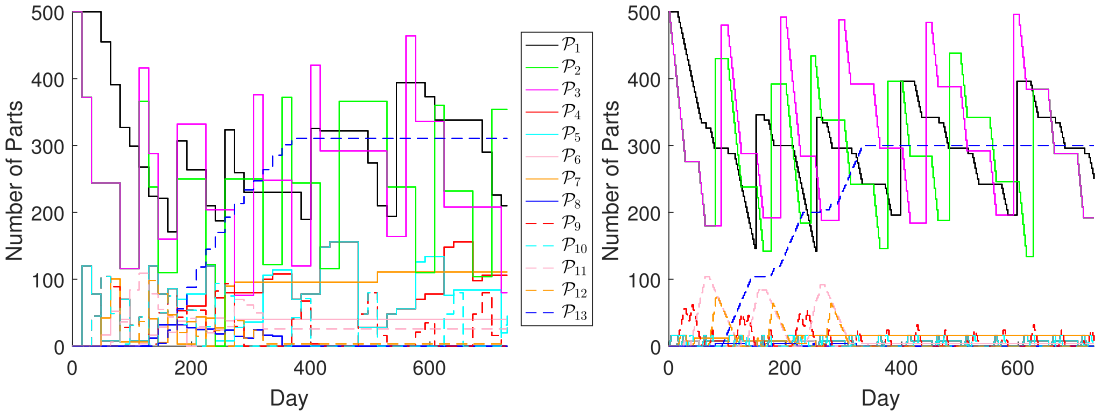

Fig. 9. The time history of the state vector in the complex pull system with transportation in Section 6.2. From left to right, the trajectories are simulated using time buckets $\Delta t = 16$ days and $\Delta t = 2$ days, respectively.

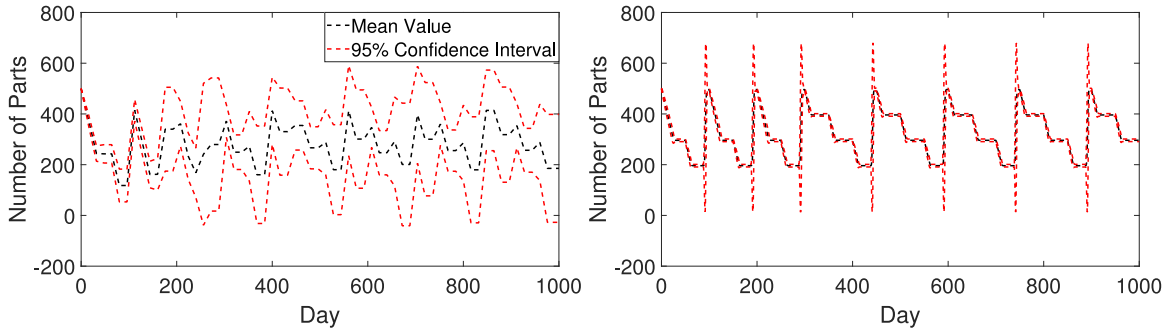

Fig. 10. The stochastic pull system with mixed orders and transportation in Section 6.2. From left to right, $\Delta t = 16$ days and $\Delta t = 0.5$ days, respectively. .

Theoretical estimation of the convergence rates can be carried out following a similar procedure associated with Theorems 1 and 2 in [3], and we leave it to a different paper. Note that in this example, we do not use the Poisson and binomial random variables in the path simulation; in other words, we only model the uncertainties introduced by the aforementioned 13 parameters.

We show the numbers of samples in Table 1. The number of required levels commonly increases as we decrease the tolerance. Nevertheless, we have overkilled the bias when the tolerances are set to be 2.5% and 0.5%. Consequently, the number of levels does not change

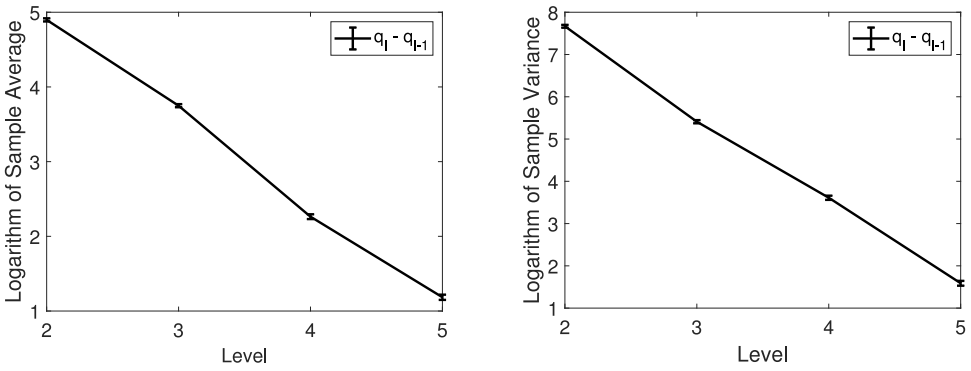

Fig. 11. Results in Section 6.3. From left to right, the graphes represent the logarithm (log_2) of the sample averages and sample variances of $q_l - q_{l-1}$, respectively. The error bars are associated with three standard deviations.

Table 1
Number of samples associated with different levels in MLMC given the tolerances in Section 6.3.

Tol (percentage)	level 0	level 1	level 2	level 3	level 4	level 5	level 6	level 7	level 8
1 (0.25%)	48,610	13,071	4248	1720	617	189	61	28	15
2 (0.5%)	11,872	3110	1085	441	164	53	16	6	2
5 (1.25%)	1700	459	145	58	21	7			
10 (2.5%)	376	103	35	14	5	2			
30 (7.5%)	43	15	5	2	1				

Table 2
The results and costs of the MLMC compared to standard MC in Section 6.3.

Tol (percentage)	result	MLMC cost (second)	MC cost (second)
1 (0.25%)	401	33.9	2470
2 (0.5%)	404	8.1	615
5 (1.25%)	402	1.1	7.5
10 (2.5%)	414	0.24	1.7
30 (7.5%)	396	0.034	0.11

when the tolerances decrease to 1.25% and 0.25%.

The results and costs of the MLMC estimator are listed in Table 2. We achieve 3.27 times acceleration when the tolerance is 30(7.5%), 7 times acceleration when the tolerances are 10(2.5%) and 5(1.25%), and 70 times acceleration when the tolerances are 2(0.5%) and 1(0.25%). The time-bucket size of the MC method is chosen as the same of the deepest level of corresponding MLMC. The rationale is stated previously in Remark 2.

6.4. uncertainty propagation using the MLMC method - pull system

In the last example, we considered uncertainty propagation in the pull system given in Section 6.2. We varied 23 parameters in the system, where $\lambda_1 - \lambda_5$ are the average production rates of corresponding processes $1 - 5$, $\lambda_6 - \lambda_{10}$ are the mean transportation rates associated with processes $6 - 10$, $x_1(t = 0)$, $x_2(t = 0)$, and $x_3(t = 0)$ are the initial inventories of $\mathcal{P}_1 - \mathcal{P}_3$, $\hat{t}_i$, $i = 1, \cdots, 5$ are the processing times of processes $1 - 5$, and $\hat{t}_i$, $i = 6, \cdots, 10$ are the transportation delays in processes $6 - 10$. The parameters are independently uniformly distributed as follows:

$$
\begin{aligned}
&\lambda_1 \sim \mathcal{U}(8, 12), \quad \lambda_2 \sim \mathcal{U}(8, 12), \quad \lambda_3 \sim \mathcal{U}(4, 6),\\
&\lambda_4 \sim \mathcal{U}(8, 12), \quad \lambda_5 \sim \mathcal{U}(1, 3),\\
&\lambda_6 \sim \mathcal{U}(7, 9), \quad \lambda_7 \sim \mathcal{U}(7, 9), \quad \lambda_8 \sim \mathcal{U}(7, 9), \quad \lambda_9 \sim \mathcal{U}(7, 9),\\
&\lambda_{10} \sim \mathcal{U}(1, 2),\\
&x_1(t = 0) \sim \mathcal{U}(800, 1200), \quad x_2(t = 0) \sim \mathcal{U}(300, 700),\\
&x_3(t = 0) \sim \mathcal{U}(800, 1200),\\
&\hat{t}_1 \sim \mathcal{U}(1, 2), \quad \hat{t}_2 \sim \mathcal{U}(1, 2), \quad \hat{t}_3 \sim \mathcal{U}(10, 20), \quad \hat{t}_4 \sim \mathcal{U}(1, 2),\\
&\hat{t}_5 \sim \mathcal{U}(10, 50),\\
&\hat{t}_6 \sim \mathcal{U}(8, 12), \quad \hat{t}_7 \sim \mathcal{U}(8, 12), \quad \hat{t}_8 \sim \mathcal{U}(8, 12),\\
&\hat{t}_9 \sim \mathcal{U}(40, 60), \quad \hat{t}_{10} \sim \mathcal{U}(8, 12),
\end{aligned}
$$

We imposed repetitive final product orders (100 per order) with 100-day intervals. We also imposed spare part orders for parts 4, 5, 6, and 7

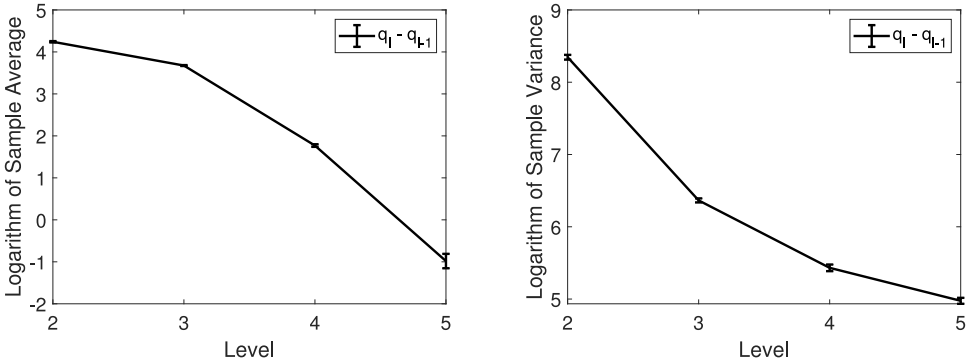

Fig. 12. The results in Section 6.4 Case 1. From left to right, the graphes represent the logarithm (log_2) of the sample averages and sample variances of $q_l - q_{l-1}$, respectively. The error bars are associated with three standard deviations. .

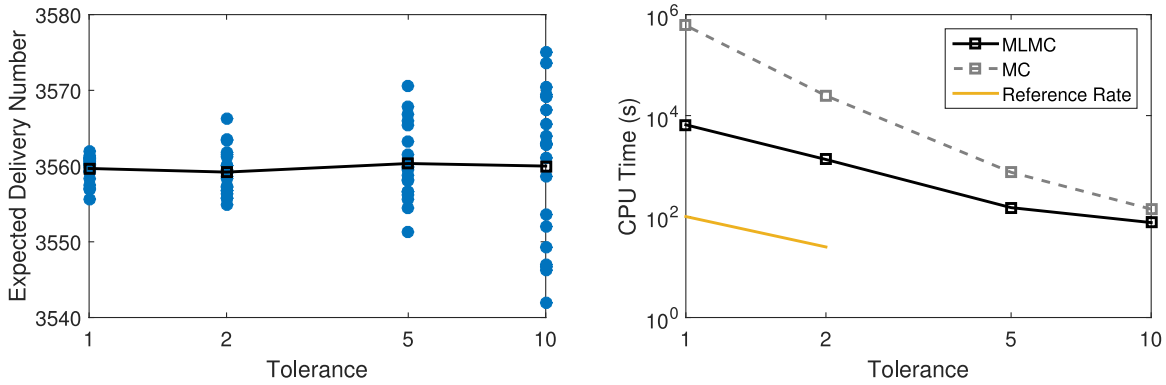

Fig. 13. The results in Section 6.4 Case 1. The left figure shows the expected delivered units of the final product in 3650 days in Example 6.4; The right figure shows the average computational cost of MLMC w.r.t. the numerical tolerances in Example 6.4. The reference is computed using the MLMC theory in Section 5.

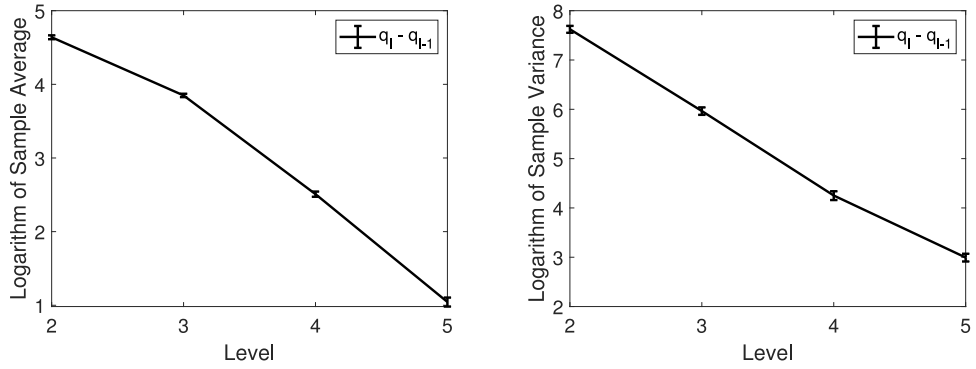

Fig. 14. The results in Section 6.4 Case 2. From left to right, the graphes represent the logarithm (log_2) of the sample averages and sample variances of $q_l - q_{l-1}$, respectively. The error bars are associated with three standard deviations. .

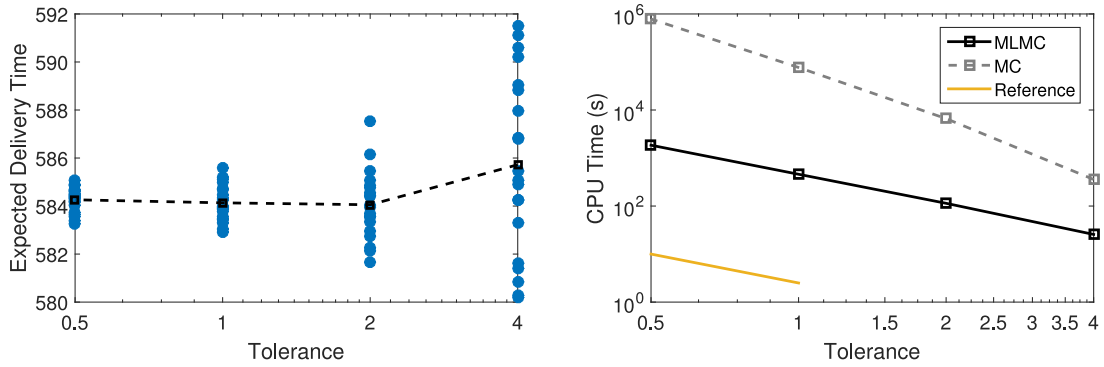

Fig. 15. The results in Section 6.4 Case 2.The left figure shows the expected delivery time of 500 units of the products in Example 6.4; The right figure shows the average computational costs of MLMC w.r.t. the numerical tolerances in Example 6.4. The reference is computed using the MLMC theory in Section 5.

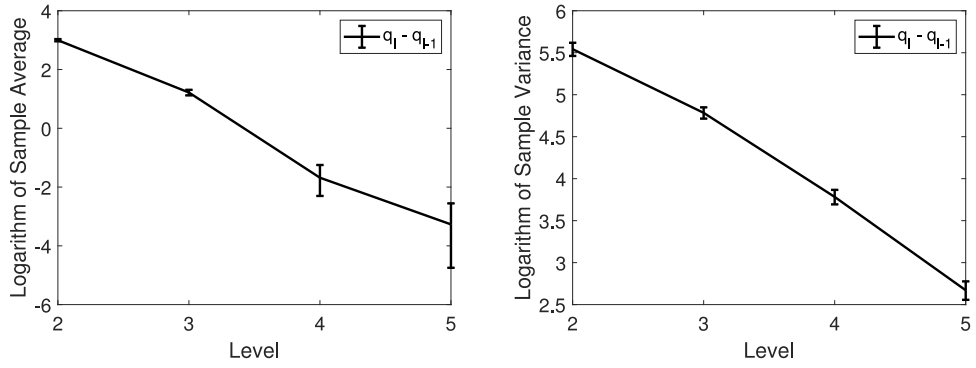

Fig. 16. The results in Section 6.4 Case 3. From left to right, the graphes represent the logarithm (log_2) of the sample averages and sample variances of $q_l - q_{l-1}$, respectively. The error bars are associated with three standard deviations. .

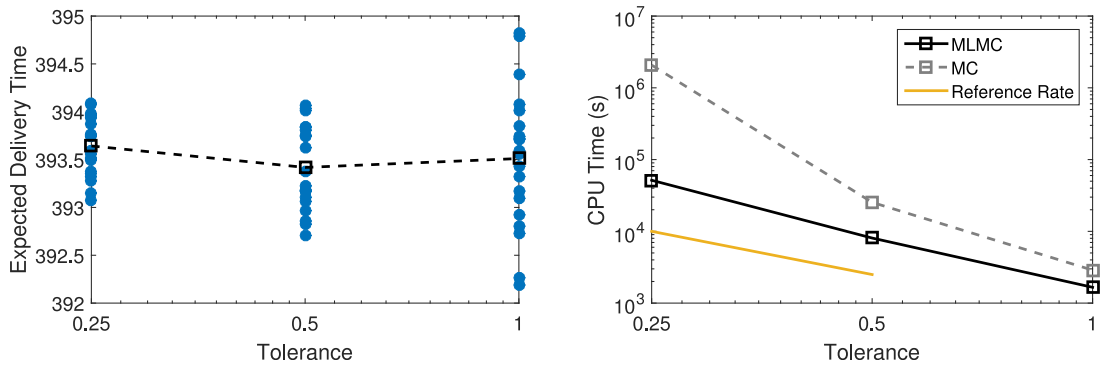

Fig. 17. The results in Section 6.4 Case 3. The left figure shows the expected delivery time of 300 units of the final products in Example 6.4; The right figure shows the average computational costs of MLMC w.r.t. the numerical tolerances in Example 6.4. The reference is computed using the MLMC theory in Section 5.

at 30 parts per order, every 30 days. We evenly split the total tolerance between the bias and statistical error (i.e., $\epsilon_b^2 = 0.5TOL^2$, $\epsilon_s^2 = 0.5TOL^2$).

6.4.1. Case 1

In Case 1, we considered the deliveries of the final products in 3650 days. We drew 10,000 independent samples of each $q_l - q_{l-1}$, where $l = 1, ..., 5$ and demonstrated the statistics of the sample mean and variance in Fig. 12 after repeating the process 100 times. The regression results are $a = 1.75 \pm 0.16$, $b = 1.10 \pm 0.05$, and $g = 1.00 \pm 0.01$ with a 99.7% confidence interval. Under parametric uncertainties, the expected deliveries of the final products in 3650 days are shown on the left in Fig. 13, where the MLMC simulation is repeated 20 times for four different values of tolerances (i.e., 1, 2, 5, and 10). The mean value of the delivery quantity converges to 3560 as we decrease the tolerance, whereas the variability of the estimator is tightly controlled by the prescribed tolerance. On the right in Fig. 13, the computational time of the MLMC method is compared with the standard MC method. The MLMC method can be several magnitudes more efficient than the standard MC method as it has a much smaller rate of growth w.r.t. the tolerance than the MC method. Note that, in this case, we did not use the Poisson and binomial random variables in the path simulation. We only modeled the uncertainties introduced by the aforementioned 23 parameters.

6.4.2. Case 2

Furthermore, we computed the expected delivery time of 500 final products. We drew 10,000 independent samples of each $q_l - q_{l-1}$, where $l = 1, ..., 5$ and demonstrated the statistics of the sample mean and variance in Fig. 14 after repeating the process 100 times. The regression results are $a = 1.21 \pm 0.02$, $b = 1.56 \pm 0.04$, and $g = 0.99 \pm 0.01$ with a 99.7% confidence interval.

Fig. 15 (on the left) shows 20 batches of MLMC simulations of the delivery time for four different values of the tolerances (i.e., 0.5, 1, 2 and 4 days). The mean value converges to 584 days, and the variability of the MLMC results is controlled rigorously by the tolerance. On the right, Fig. 15 compares the computational time of the MLMC estimator with the standard MC method. Again, the MLMC method is several orders of magnitude faster than the standard MC method. More specifically, it is 10 times faster than the MC method when the tolerance is 4 days. This factor grows to 100 as we reduce the tolerance to 0.5 days. In this case, we did not use the Poisson and binomial random variables in the path simulation; in other words, we only modeled the uncertainties introduced by the aforementioned 23 parameters.

6.4.3. Case 3

Finally, we considered both parametric uncertainties and uncertainties driven by the Poisson and binomial random variables in the path simulations. Specifically, the number of processes occurring in any time bucket is a Poisson random variable (11), and the number of items produced is an stochastic process related to a binomial distribution (12). We computed the expected delivery time of 300 final products using the MLMC method. We chose $\Delta t = 5$ days as the coarsest level. We drew 10,000 independent samples of each $q_l - q_{l-1}$, where $l = 1, ..., 5$ and demonstrated the statistics of the sample mean and variance in Fig. 16 after repeating the process 100 times. The regression results are $a = 2.26 \pm 0.39$, $b = 0.97 \pm 0.03$, and $g = 0.99 \pm 0.01$ with a 99.7% confidence interval. On the left in Fig. 17, the results of 20 runs of the MLMC method are displayed against the tolerances. The average delivery time converges to 393.6 days. Fig. 17 (on the right) shows the average computational costs of the MLMC method w.r.t. the tolerances. For a tolerance smaller than 1, the MLMC method is significantly advantageous to the standard MC method because the multilevel complexity grows much more slowly than the MC method.

7. Conclusion

We have presented a multilevel uncertainty propagation framework using a time-bucket method of simulating manufacturing supply chains. We incorporated several distinctive dynamical features of supply chain simulations, including limited capacities, push and pull productions, transportation, inventory replenishment, and priority productions, into the leap methods that were previously used to approximate the DES of chemical and biochemical systems. The time buckets naturally offer a hierarchy of models that can be combined with the MLMC method to accelerate the propagation of uncertainties in a supply chain network. We demonstrated more than 10 times the acceleration using our approach compared to the standard MC method using several manufacturing supply chain examples. Specifically, using DES to model supply chain management and manufacturing processes, a medium-sized model can easily take up to $0.5 - 1$ hour to complete. To perform statistically meaningful what-if analyses or optimizations, it is reasonable to expect that a simulation model may be run hundreds of times to generate one desired result, which will take days to complete. With the acceleration of the proposed approach, the running time can be reduced from hundreds of hours to tens of hours. Thus, the analysis now can be complete within one day, instead of one week using the traditional MC approach. Considering future work, we note that the framework of combining time buckets and the MLMC method can be applied to the agent-based [65] and continuous modeling [19] of a supply chain to achieve efficient uncertainty propagation. The approach can also benefit decision-making in a supply chain, for example, regarding circular economy modeling and green supply chain management, as it accelerates the prediction of uncertainties using a discrete supply chain model.

Declaration of Competing Interest

The authors declare that they have no known competing financial

interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to acknowledge support from United Technologies Research Center through the innovation pipeline program and the capability program of the systems department. We thank Thomas Frewen and Bob LaBarre for their valuable comments. Finally, we thank the anonymous referees for their careful reading and constructive comments that substantially improved the quality of the draft.

References

[1] Alfieri A, Brandimarte P. Object-oriented modeling and simulation of integrated production/distribution systems. Comput Integr Manuf Syst 1997;10:261–6.
[2] Anderson DF. Incorporating postleap checks in tau-leaping. J Chem Phys 2008;128:54103.
[3] Anderson DF, Higham DJ. Multilevel monte carlo for continuous time markov chains, with applications in biochemical kinetics. Multiscale Model Simul 2012;10:146–79.
[4] Angulo A, Nachtmann H, Waller MA. Supply chain information sharing in a vendor managed inventory partnership. J Bus Logist 2004;25:101–20.
[5] Baldwin R. Global supply chains: Why they emerged, why they matter, and where they are going. 2012. CEPR Discussion Papers, 9103, C.E.P.R. Discussion Papers.
[6] Bayati B, Chatelain P, Koumoutsakos P. D-Leaping: accelerating stochastic simulation algorithms for reactions with delays. J Comput Phys 2009;228:5908–16.
[7] Beamon BM, Chen VC. Performance analysis of conjoined supply chains. Int J Prod Res 2001;39:3195–218.
[8] Beck J, Dia BM, Espath LF, Long Q, Tempone R. Fast bayesian experimental design: laplace-based importance sampling for the expected information gain. Comput Methods Appl Mech Eng 2018;334:523–53.
[9] Bhaskaran S. Simulation analysis of a manufacturing supply chain. Decis Sci 1998;29:633–57.
[10] Biswas S, Narahari Y. Object oriented modeling and decision support for supply chains. Eur J Oper Res 2004;153:704–26.
[11] Boulaksil Y. Safety stock placement in supply chains with demand forecast updates. Oper Res Perspect 2016;3:27–31.
[12] Byrne P, Heavey C. The impact of information sharing and forecasting in capacitated industrial supply chains: a case study. Int J Prod Econ 2006;103:420–37.
[13] Cao Y, Gillespie DT, Petzold LR. Efficient step size selection for the tau-leaping simulation method. J Chem Phys 2006;124:44109.
[14] Cao Y, Petzold LR, Rathinam M, Gillespie DT. The numerical stability of leaping methods for stochastic simulation of chemically reacting systems. J Chem Phys 2004;121:12169–78.
[15] Ceroni JA, Nof SY. A workflow model based on parallelism for distributed organizations. J Intell Manuf 2002;13:439–61.
[16] Chatterjee A, Vlachos DG, Katsoulakis MA. Binomial distribution based τ-leap accelerated stochastic simulation. J Chem Phys 2005;122:24112.
[17] Collier N, Haji-Ali A-L, Nobile F, von Schwerin E, Tempone R. A continuation multilevel monte carlo algorithm. BIT Num Math 2015;55:399–432.
[18] Daniel JSR, Rajendran C. A simulation-based genetic algorithm for inventory optimization in a serial supply chain. Int Trans Oper Res 2005;12:101–27.
[19] D'Apice C, Gottlich S, Herty M, Piccoli B. Modeling, simulation, and optimization of supply chains: a continuous approach, vol. 121. SIAM; 2010.
[20] Deleris LA, Erhun F. Risk management in supply networks using monte-carlo simulation. Proceedings of the 37th winter simulation conference. 2005. p. 1643–9.
[21] Ding H, Benyoucef L, Xie X. A simulation optimization methodology for supplier selection problem. Int J Computer Integr Manuf 2005;18:210–24.
[22] Ding H, Benyoucef L, Xie X. A simulation-based multi-objective genetic algorithm approach for networked enterprises optimization. Eng Appl Artif Intell 2006;19:609–23.
[23] Dong M, Chen FF. The impacts of component commonality on integrated supply chain network performance: a state and resource-based simulation study. Int J Adv Manuf Technol 2005;27:397–406.
[24] Dong M, Chen FF. Performance modeling and analysis of integrated logistic chains: an analytic framework. Eur J Oper Res 2005;162:83–98.
[25] Fleisch E, Tellkamp C. Inventory inaccuracy and supply chain performance: asimulation study of a retail supply chain. Int J Prod Econ 2005;95:373–85.
[26] Fleischmann M, Van Nunen JA, Gräve B. Integrating closed-loop supply chains and spare-parts management at IBM. Interfaces 2003;33:44–56.
[27] Fu MC, Bayraksan G, Henderson SG, Nelson BL, Powell WB, Ryzhov IO, et al. Simulation optimization: a panel on the state of the art in research and practice. Proceedings of the 46th winter simulation conference. 2014. p. 3696–706.
[28] Fujimoto RM. Parallel discrete event simulation. Commun ACM 1990;33:30–53.
[29] Fujimoto RM, Bagrodia R, Bryant RE, Chandy KM, Jefferson D, Misra J, et al. Parallel discrete event simulation: the making of a field. Proceedings of the 49th winter simulation conference. 2017. p. 262–91.
[30] Garrido JM. Practical Process simulation using Object-Oriented Techniques and C++. Artech House; 1998.
[31] Geraghty J, Heavey C. A comparison of hybrid push/pull and conwip/pull production inventory control policies. Int J Prod Econ 2004;91:75–90.
[32] Ghrayeb O, Phojanamongkolkij N, Tan BA. A hybrid push/pull system in assemble-to-order manufacturing environment. J Intell Manuf 2009;20:379.
[33] Giles MB. Multilevel monte carlo path simulation. Oper Res 2008;56:607–17.
[34] Giles MB. Multilevel monte carlo methods. Acta Numerica 2015;24:259–328.
[35] Gillespie DT. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. J Comput Phys 1976;22:403–34.
[36] Gillespie DT. Approximate accelerated stochastic simulation of chemically reacting systems. J Chem Phys 2001;115:1716–33.
[37] Grossmann IE. Enterprise-wide optimization: a new frontier in process systems engineering. AlChE J 2005;51:1846–57.
[38] Hodgson TJ, Wang D. Optimal hybrid push/pull control strategies for a parallel multistage system: part i. Int J Prod Res 1991;29:1279–87.
[39] Hodgson TJ, Wang D. Optimal hybrid push/pull control strategies for a parallel multistage system: part ii. Int J Prod Res 1991;29:1453–60.
[40] Jahangirian M, Eldabi T, Naseer A, Stergioulas LK, Young T. Simulation in manufacturing and business: a review. Eur J Oper Res 2010;203:1–13.
[41] Jain S, Ervin E. Evaluation of supply chain business process improvements using simulation. Int J Simul Process Model 2005;1:138–49.
[42] Juan AA, Faulin J, Grasman SE, Rabe M, Figueira G. A review of simheuristics: extending metaheuristics to deal with stochastic combinatorial optimization problems. Oper Res Perspect 2015;2:62–72.
[43] Jung JY, Blau G, Pekny JF, Reklaitis GV, Eversdyk D. A simulation based optimization approach to supply chain management under demand uncertainty. Comput Chem Eng 2004;28:2087–106.
[44] Kapuscinski R, Tayur S. Optimal policies and simulation-based optimization for capacitated production inventory systems. Quantitative models for supply chain management. Springer; 1999. p. 7–40.
[45] Kim J, Realff MJ, Lee JH. Optimal design and global sensitivity analysis of biomass supply chain networks for biofuels under uncertainty. Comput Chem Eng 2011;35:1738–51.
[46] Kleijnen JP, Ridder AA, Rubinstein RY. Variance reduction techniques in monte carlo methods. Encyclopedia of operations research and management science. Springer; 2013. p. 1598–610.
[47] Law AM. Simulation Modeling and Analysis. McGraw-Hill Higher Education; 2014.
[48] Law AM, Kelton WD, Kelton WD. Simulation Modeling and Analysis. 2. McGraw-Hill New York; 1991.
[49] Lee C-Y, Lee D. An efficient method for solving a correlated multi-item inventory system. Oper Res Perspect 2018;5:13–21.
[50] Li S. Marked event method in discrete event simulation. Proceedings of the 21st winter simulation conference. 1989. p. 719–28.
[51] Lu H, Liu X, Pang W, Ye WH, Wei BS. Modeling and simulation of aircraft assembly line based on quest. Advanced materials research. vol. 569. 2012. p. 666–9.
[52] Maravelias CT, Grossmann IE. New general continuous-time state task network formulation for short-term scheduling of multipurpose batch plants. Ind Eng Chem Res 2003;42:3056–74.
[53] Montevechi JAB, de Pinho AF, Leal F, Marins FAS. Application of design of experiments on the simulation of a process in an automotive industry. Proceedings of the 39th winter simulation conference. 2007. p. 1601–9.
[54] Moraes A, Tempone R, Vilanova P. Hybrid chernoff tau-leap. Multiscale Model Simul 2014;12:581–615.
[55] Moraes A, Tempone R, Vilanova P. Multilevel hybrid chernoff tau-leap. BIT Num Maths 2016;56:189–239.
[56] Mustafee N, Katsaliaki K, Taylor SJ. A review of literature in distributed supply chain simulation. Proceedings of the 46th winter simulation conference. 2014. p. 2872–83.
[57] Novak SY. Extreme Value Methods with Applications to Finance. London: CRC/Chapman & Hall/Taylor & Francis; 2011.
[58] Ohno T. Toyota Production System: Beyond Large-Scale Production. CRC Press; 1988.
[59] Ptak C, Smith C. Orlicky's Material Requirements Planning. Third ed McGraw-Hill Education; 1994.
[60] Rosenblatt MJ, Roll Y, Vered Zyser D. A combined optimization and simulation approach for designing automated storage/retrieval systems. IIE Trans 1993;25:40–50.
[61] Saridakis G, . Papaioannou G, Dani S. Introduction to the special issue on supply chain decision making in times of unrest, instability and change. Oper Res Perspect 2018;5:80–1.
[62] Schmitt AJ, Singh M. Quantifying supply chain disruption risk using monte carlo and discrete-event simulation. Proceedings of the 41st winter simulation conference. 2009. p. 1237–48.
[63] Shapiro JF. Modeling the Supply Chain. 2. Cengage Learning; 2006.
[64] Simatupang TM, Sridharan R. The collaborative supply chain. Int J Logis Manag 2002;13:15–30.
[65] Swaminathan JM, Smith SF, Sadeh NM. Modeling supply chain dynamics: a multiagent approach. Decis Sci 1998;29:607–32.
[66] Tako AA, Robinson S. The application of discrete event simulation and system dynamics in the logistics and supply chain context. Decis Support Syst 2012;52:802–15.
[67] Taylor SJ. Distributed simulation: state-of-the-art and potential for operational research. Eur J Oper Res 2019;273:1–19.
[68] Thierry C, Thomas A, Bel G. Simulation for Supply Chain Management. Wiley; 2008.
[69] Wong WP, Jaruphongsa W, Lee LH. Supply chain performance measurement system: a monte carlo dea-based approach. Int J Ind Syst Eng 2008;3:162–88.
[70] Yan Y, Wang G. A job shop scheduling approach based on simulation optimization. Industrial Engineering and Engineering Management, 2007 IEEE International Conference on. IEEE; 2007. p. 1816–22.
[71] You F, Grossmann IE. Mixed-integer nonlinear programming models and algorithms for large-scale supply chain design with stochastic inventory management. Ind Eng Chem Res 2008;47:7802–17.
[72] Ziarnetzky T, Mönch L, Biele A. Simulation of low-volume mixed model assembly lines: modeling aspects and case study. Proceedings of the 46th winter simulation conference. 2014. p. 2101–12.