Efficient Simulation Method for General Assembly Systems With Material Handling Based on Aggregated Event-Scheduling

Yanjia Zhao, *Student Member, IEEE*, Chao-Bo Yan, *Student Member, IEEE*, Qianchuan Zhao, *Senior Member, IEEE*, Ningjian Huang, Jingshan Li, *Senior Member, IEEE*, and Xiaohong Guan, *Fellow, IEEE*

***Abstract*—Performance evaluation of complex manufacturing systems is challenging due to many factors such as system complexity, parameter uncertainties, problem size, just to name a few. In many cases when a system is too complex to model using mathematical formulas, simulation is used as an effective alternative to conduct system analysis. A manufacturing system is a good example of such cases where both system performance and system complexity are greatly impacted by material handling (MH) strategy, management, and operational control. In this paper, we study vehicle general assembly (GA) system with MH, and focus on developing an efficient simulation method for modeling and analysis where traditional simulation methods may suffer from computation intensity. Making use of the partial system decomposability, we introduce an aggregated event-scheduling simulation method with two-level framework. A dividing mechanism with boundary conditions is employed in top-level simulation to divide the global event list into small sizes. A timing-focuses strategy based on max-plus algebra is applied in bottom-level local simulation to further reduce local event lists. With this new method it is possible to mimic real production systems fast and accurately within a reasonable computational time frame. The effectiveness and efficiency of the new simulation method are validated through experimental results.**

***Note to Practitioners*—In manufacturing systems with MH (e.g., GA lines in automotive industry), MH has great impact on system performance. It also increases system complexity and in turn causes difficulties and challenges in modeling and simulation. In this paper, we focus on developing an efficient simulation approach for a GA system with MH. An aggregated event-scheduling simulation method with two-level framework (a dividing mechanism with boundary conditions for top level and a timing focus strategy based on max-plus algebra for bottom level) is introduced to reduce the size of event list. With this new method it is possible to mimic real production fast and accurately. The experimental results suggest that the method provides an efficient way for simulation modeling and analysis of assembly system with MH.**

***Index Terms*—Aggregated event-scheduling, discrete-event dynamic system, general assembly (GA) systems, material handling (MH), simulation.**

I. Introduction

MATERIAL HANDLING (MH) is an important element in vehicle production [4]. In modern automotive assembly plants, MH system and assembly processes are closely related. Timely and accurate delivery of necessary material to the assembly line is essential to achieve high productivity and quality. It is claimed in [5] that 20%–50% of the manufacturing costs may be related to MH. Therefore, efficient and precise analysis of system performance in general assembly (GA) lines with MH is necessary and important.

In an automotive assembly plant, a GA line is typically structured as follows (see Fig. 1) [1]–[3].

A *main production line* is composed of a series of *sections*, with each pair being separated by *in-process buffers* (or, *section buffers*). Typically, such sections include trims, chassis, door line, final assembly, and inspection. Within each section, consecutive serial *stations* are connected through a paced-moving conveyor [2], [3]. The sections are asynchronous in nature so that each is moving independently. However, the stations within each section are synchronized so that all stations in this section stop working if any of them stops. In addition, to increase productivity, some *component*s (e.g., doors, powertrain, suspension, etc.) may be preassembled in the *subassembly lines* before merged into the main line. The *vehicles* flow into the system on the moving conveyors, passing through all the stations. Each station has *lineside buffers* supplying parts needed for assembly. At each station, operators load the required parts from the lineside buffers and assemble them on the vehicles. The parts to replenish the lineside buffers are delivered by the MH workers from central docking area based on the production schedule and delivery policy using dollies, tuggers or forklifts.

As one can see, in addition to machine breakdowns, a station could also suffer from blockages and starvations due to interactions among up- or downstream stations. Moreover, a station can also be starved and stop working if the necessary parts in the lineside buffers are not available. Clearly, larger inventories in the lineside buffers could keep production from being starved

Manuscript received December 29, 2008; revised May 31, 2009. This paper was recommended for publication by Associate Editor A. Patchong and Editor M. Zhou upon evaluation of the reviewers' comments. This work was presented in part at the Fourth IEEE Conference on Automation Science and Engineering, 2008. This paper was supported by a contract between Tsinghua University and General Motors Corporation. The work of Q. Zhao and X. Guan was supported in part by the National Science Foundation of China (NSFC) under Grant 60574067, Grant 60736027, and Grant 60721003, in part by the 863 High Tech Development Plan (2007AA04Z154), and in part by the Program of Introducing Talents of Discipline to Universities (the 111 International Collaboration Project of China).

Y. Zhao, C.-B. Yan, Q. Zhao, and X. Guan are with the Center for Intelligent and Networked Systems, Department of Automation, TNList, Tsinghua University, Beijing 100084, China (e-mail: zhaoqc@tsinghua.edu.cn).

N. Huang is with the Research and Development Center, General Motors Corporation, Warren, MI 48090 USA.

J. Li is with the Department of Electrical and Computer Engineering and the Center for Manufacturing, University of Kentucky, Lexington, KY 40506 USA.

Digital Object Identifier 10.1109/TASE.2009.2034135

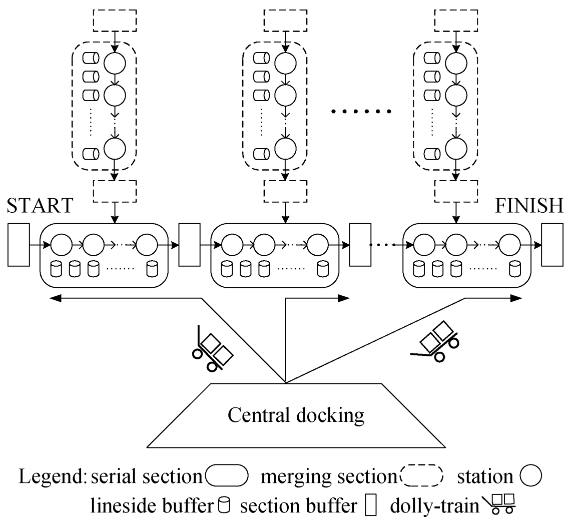

START
FINISH
Central docking
Legend: serial section ⬭ merging section station ○
lineside buffer section buffer ▯ dolly-train

Fig. 1. Sketched GA system with MH.

due to parts and make the analysis of the GA lines simple (by omitting the MH). However, this is not pursuable due to cost, quality, and floor space constraints. Lean lineside inventories are more desirable. In this case, how to ensure a smooth production with MH system in GA lines is an important question.

Analysis of production lines has attracted substantial attention from both research and practice during the last forty years and significant improvement has been achieved (see, for instance, review [9] and monographs [1], [6]). Most of them emphasize on serial lines and assembly systems. It is typically assumed that MH is not an issue or has minimum impact so that the line always has sufficient materials and will not be starved by parts. In some cases, the effect of part shortage is approximated by reducing station reliability. However, such approximation does not fully address the interactions between material delivery and vehicle assembly.

Including MH into the analysis is challenging, since in addition to the traditional difficulties, such as randomness and nonlinearity of production, interactions between stations, asynchronous and synchronous natures between and within sections, etc., more difficulties arise due to the inconsistency and coupling between MH and GA. Specifically, the inconsistency may include the following.

1) Assembly (or production) process is carried out in the time scale of cycles (e.g., minutes), while material delivery is based on shipping schedule in the time scale of many cycles (e.g., hours).
2) Assembly process is implemented in the unit of individual vehicle, but MH transports parts in batches (e.g., carts, boxes).
3) Moreover, in the assembly process, each vehicle may need multiple types of parts with various quantities at each station, depending on the customer options.

The coupling between MH and GA exists since:

1) The lack of one particular part may result in the stoppage of the station and the whole section, and propagate to the sections up- and downstream.
2) The MH worker may deliver parts to many stations (may not belong to a single section), which introduces global correlations among those stations and sections.
3) The change of control strategy in material delivery may interrupt the assembly process and introduce more transient phenomenon. Therefore, the traditional steady-state analysis may not be able to address.

Due to the above complexities and coupling issues, analytical evaluation of system performance for GA line with MH seems impossible. Therefore, simulation approach is pursued to analyze the system behavior. However, the traditional discrete-event simulation may suffer from long simulation time and computation intensity for large scales. Thus, developing an efficient simulation method to provide accurate and quick solutions of performance evaluation is of importance. The goal of this paper is intended to contribute to this end.

The main contribution of this paper is the development of a new efficient simulation method for GA lines with MH in automotive manufacturing. Specifically, a method based on aggregated even-scheduling is proposed with the following salient features.

1) Dividing mechanism. By making use of partial decomposability, a novel dividing mechanism with boundary conditions is introduced to aggregate the simulation for each section into rounds.
2) Reduced event list. With a timing-focused simulation strategy based on max-plus algebra, many redundant events are omitted and only a small key-event list is kept in order to improve simulation efficiency.
3) Potential parallelism. It is possible to extend the new method for parallel and distributed simulation. Since the simulation framework is divided into two levels, individual sections of the GA line can be simulated in parallel and synchronized through the global system states and events.

With these salient features, the aggregated event-scheduling simulation method is possible to simulate actual production systems accurately enough with reasonable computational times. Numerical experiments and case studies suggest that such method results in efficient and effective analysis of GA line with MH. Additionally, this method is also applicable to other manufacturing systems, such as the disassembly line, and systems with fork/join structures, such as parallel processing systems and distributed replicated database systems [48], [49].

The remaining part of this paper is organized as follows. Section II presents literature reviews on analysis and simulation approaches for production systems and discrete-event dynamic systems. Section III provides mathematical descriptions of the system. Section IV presents the aggregated event-scheduling simulation method. Numerical results are illustrated in Section V to demonstrate the effectiveness and efficiency of the new method and followed by Conclusions.

II. Literature Review

A General Assembly (GA) line is a typical type of manufacturing system. As one of the few ways that wealth is created, manufacturing system has attracted much research interest over the past two decades, which can be roughly categorized into two classes: the system design problem and the scheduling and real-time decision making problem. The MH system usually takes charge of the delivery of both raw parts and finished parts, and it consists of loading and unloading mechanisms, transfer mechanisms, and internal storage facilities, etc. [4]. MH is widely required in various segments of manufacturing industries, and it is believed that the design of MH systems has great impact on the overall manufacturing process cost and efficiency [5]. The research of a MH system can be similarly divided into two aspects: the design problem and the operation problem, as surveyed in [7].

Although there is substantial research on the GA lines and MH systems individually, it still lacks of analytical results for the problem combining these two systems together in a specific way. Since the GA line with MH system does not satisfy Markovian assumptions for processing time and routing probabilities in queueing network, it is not a product form queueing network which allows analytical solutions. Even more general Markov chain and Semi Markov Chain [8] models could not be applied directly to describe this system and obtain analytical solutions, since the time intervals between events (state changing) do not follow the exponential distribution and sections are asynchronous with different cycle times. Approximation and empirical approaches such as aggregations [1], [9] and decomposition [6], may not provide accurate and detailed information. Furthermore, when dealing with production control laws and MH strategies, there are no analytical tools available. Thus, simulation is one of the most important approaches available in this case. To this end, this paper focuses on developing an efficient simulation approach for the GA line with MH illustrated in Fig. 1.

Simulation has been widely and successfully applied in the design and optimization of DEDS, especially manufacturing systems [8], [10], [12], [13]. For example, [14]–[20] develop simulation models for different types of manufacturing systems to optimize the design and control strategies. References [2], [3], [21]–[24] apply simulations for automotive production lines to investigate system parameters and test various hypotheses. References [25]–[29] simulate MH systems of flexible manufacturing systems, such as semiconductor fabs, to evaluate various designs and scheduling rules. Reference [38] investigates optimal dispatching policies in MH systems of GA lines based on the efficient simulation method developed in this paper. These successful applications demonstrate the advantages of simulations, i.e., flexibility, time compression, physical scaling, and risk avoidance [10], [11].

Conceptually, all these discrete-event simulation models mentioned above can be exactly formulated as the Generalized Semi-Markov Process (GSMP) mathematically. The evolution of a GSMP is governed by its state transition function. Different ways of doing simulation can be viewed as different methods of updating system states. Since the state space is extremely large for the system under consideration, one has to avoid explicitly storing the state transition function into a table for the GSMP as in event scheduling scheme [10], [12], instead one has to define it implicitly by exploring the system structure so that efficient simulation can be carried out. Below we will focus on existing simulation approaches of DEDS and point out their limitations when applied directly to the system we study. Despite great diversities of real-world systems, the *event scheduling scheme* (also known as the *next-event time-advance approach*) [10], [12] provides a general simulation approach for DEDS [8]. In this approach, a simulation clock and an event list are introduced. A timing routine is invoked to determine which event in the event list will occur next. The simulation clock is advanced to the time when that event happens and an event routine is invoked to update the system state and to generate future events. The procedure is repeated until the stopping condition is eventually satisfied. Dozens of successful software packages have been developed based on this approach, such as Arena, AutoMod, and ProModel, etc. However, handling an event list with huge size makes this approach extremely difficult to be applied for large-scale practical problems.

Unlike the event scheduling approach, the max-plus algebra (min-max algebra) introduces another approach to model DEDS [30]–[32]. It studies system behaviors through recursion formulas based on the timing-logic and can thus handle a large number of events efficiently. However, this approach is proper to deal with the systems with deterministic events together with the decision-free requirement. It is not applicable for the systems with random failures and state transitions depending on both upstream and downstream stations within a section. Therefore, the max-plus algebra approach could not be applied directly to simulate the GA line with MH.

Parallel and distributed simulation is another relevant research area. Four synchronization mechanisms: conservative, optimistic, hybrid, and adaptive, are widely investigated, such as in [33]–[35]; performance analysis and applications of these mechanism and corresponding protocols are studied, such as in [44]–[47]. An obvious benefit of parallel simulation is that computational times can be reduced by dividing a large simulation task into many subtasks that can be executed concurrently. For discrete-event systems, many advances have been made (see e.g., [39], [40], and [42]), which provide us many insights including cutting the system into pieces at buffers (queues), introduction of asynchronous protocols to develop our simulation strategy. To successfully harvest the benefit of parallel simulation, one has to carefully divide the simulation task so that the overhead caused by interaction among subtasks[1] is not a major problem. For discrete-event systems with relatively simple structure, the division is not very hard: for example, for the FCFS queueing networks with infinite buffers, one can follow [40] to regard the simulation of each node (every queueing) as a subtask and handle the interaction among nodes by inserting arrival events to the queue of every node properly.

However, our problem is more challenging. Comparing with existing parallel and distributed discrete-event simulation problems [39], [42], [43], the simulation of the GA line with MH is much complicated. The stations in each section are strongly coupled and synchronized by the paced-moving conveyor in the GA line; the MH system introduces global coupling among sections since drivers are shared by difference sections when supplying parts. These challenges make the division of the simulation task and the design of interaction mechanism a nontrivial problem. Detailed discussion can be found in Section IV-D.

[1]Since some of the events have to be synchronized at certain points.

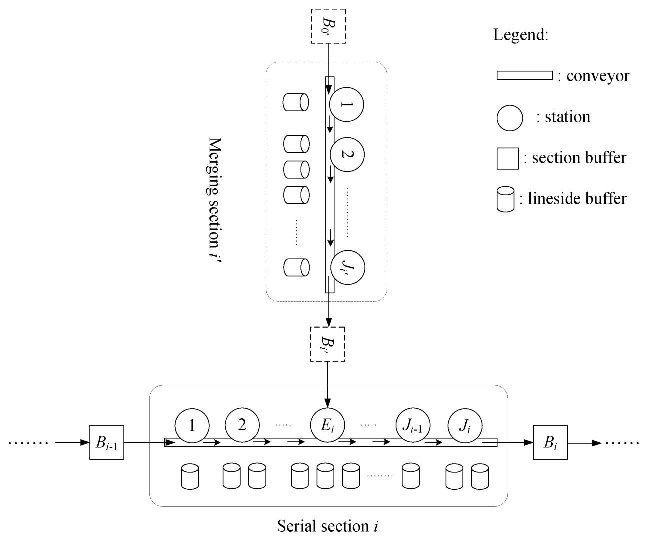


Fig. 2. Detailed structure of one section in the GA line.

To summarize, it is desirable to develop a new effective simulation method for the GA line with MH systems by investigating its structure characteristics. This research is with significant theoretical importance and broad application potentials.

III. System Descriptions

This section formally presents the GA line with MH under consideration.

A. Overview of the System

The following notation is firstly introduced to denote the GA line with MH system. Sections in the main line are named as *serial sections*, and sections in the subassembly line are named as *merging sections*. Suppose the system consists of I serial sections and I merging sections

$$\mathcal{I}_s \cup \mathcal{I}_m = \mathcal{I}, \quad \mathcal{I}_s \cap \mathcal{I}_m = \emptyset \tag{1}$$
$$|\mathcal{I}_s| = |\mathcal{I}_m| = I \tag{2}$$

where $\mathcal{I}$ denotes the set of sections in the GA line; $\mathcal{I}_s$ and $\mathcal{I}_m$ denote the sets of serial sections and merging sections in the GA line, respectively. Hereafter, we suppose $\mathcal{I}_s = \{1, 2, \ldots, I\}$, $\mathcal{I}_m = \{1', 2', \ldots, I'\}$.

Based on the sketched layout of the entire system illustrated in Fig. 1, we redraw detailed structure of one section in the GA line in the figure below. As Fig. 2 shows, we use the following methods to indicate the connection relationships of the GA line: the *input buffer* of serial section i, i.e., the section buffer before serial section i, is denoted as buffer $i-1$; the *output buffer* of serial section i, i.e., the section buffer after serial section i, is denoted as buffer i; the merging section joining into serial section i is indexed as i', which is also used to indicate the index of section buffers in between.

Section $i, \forall i \in \mathcal{I}$, consists of a paced-moving conveyor and a series of consecutive stations. Vehicles are transferred on the conveyor step by step, going through all the stations where various parts are assembled onto the vehicle. The total number of stations in section i is J_i, i.e.,

$$|\mathcal{J}_i| = J_i \tag{3}$$

where $\mathcal{J}_i = \{1, 2, \ldots, J_i\}$ denotes the set of stations in section i. Note that we use $j = 1, 2, \ldots, J_i$ to indicate the consecutive order of stations in section i. For example, index 1 denotes the first station and index J_i is the last one in section i.

Each driver in the MH system takes charge of a given set of lineside buffers and there are no overlaps

$$\mathcal{L} = \cup_{i \in \mathcal{I}} \cup_{j \in \mathcal{J}_i} \mathcal{L}_{ij} \tag{4}$$
$$\mathcal{L}(d) \cap \mathcal{L}(d') = \emptyset, \quad \forall d \neq d', d, d' \in \mathcal{D} \tag{5}$$
$$\cup_{d \in \mathcal{D}} \mathcal{L}(d) = \mathcal{L} \tag{6}$$

where $\mathcal{L}$ denotes the set of lineside buffers in the entire system; $\mathcal{L}_{ij}$ is the set of lineside buffers belonging to station j in section i; $\mathcal{D}$ indicates the set of drivers of the MH system; $\mathcal{L}(d)$ is the set of lineside buffers supplied by driver d. Note that it is the current setting in practical systems that responsibilities of drivers are fixed and distinguished. However, the simulation method developed in this paper could also handle the scenario if all drivers are pooled and supply all lineside buffers based on availability. Numerical test for this alternative setting is shown in the online technical report [50]. To simplify the explanation, we make the following assumptions based on literature traditions and industry requirements.

1) Buffers have finite capacities.
2) All stations are unreliable. The time between failures and the time to repair for each station are exponentially distributed. The exponential reliabilities are widely assumed in studies of manufacturing systems [1], [6]; the effectiveness of this assumption is evaluated by [36]. Note that the simulation method developed in this paper is also applicable to other distributions without memoryless property by augmenting the station state with accumulated working time and repairing time.
3) The *cycle time* of station j in section i, i.e., the time necessary to process a vehicle by a station, is deterministic and denoted as τ_{ij}. Stations in the same section have the same constant cycle time, i.e.,

$$\tau_{ij} = \tau_{i,'} = \tau_i, \forall j, j' \in \mathcal{J}_i \tag{7}$$

where τ_i is defined as the cycle time of section i. Sections may have different cycle times in general, i.e.,

$$\tau_i \neq \tau_{i'}, \forall i, i' \in \mathcal{I}, i \neq i'. \tag{8}$$

Thus, dynamics of sections are asynchronous.
4) The conveyor will move as long as there is at least one vehicle in the section, stations have no failures, and the output buffer is not full.
5) The vehicle at the beginning buffer, B_0 and B_0's, are always ready and the finished good buffer B_I is infinite. The numbers of all parts in the central docking area are infinite.

B. Mathematical Description of Stations

For station j in section $i, \forall i \in \mathcal{I}, j \in \mathcal{J}_i$, it has three states at time t, *up*, *down* and *idle*, which is denoted as $s_{ij}(t)$. The station is *up*, if it is busy with a vehicle under assembling at time t, denoted as $s_{ij}(t) = 1$; the station is *down*, if it is broken down and in repair at time t, denoted as $s_{ij}(t) = -1$; otherwise the station is *idle*, denoted as $s_{ij}(t) = 0$. The idle state contains three cases: being empty, or holding a vehicle to output, or starved by parts at some lineside buffer belonging to it.

The time duration of the station being *up*, i.e., the accumulated working time, is defined as *uptime*; the time duration of the station in failure is defined as *downtime*. The stochastic reliability model for the stations in the system is depicted with exponential distributions of uptime and downtime. The probability density functions of the up- and downtime of station j in section i at time t are given by

$$f_{up} = \lambda_{ij} e^{-\lambda_{ij} t}, t \geq 0 \tag{9}$$

$$f_{down} = \mu_{ij} e^{-\mu_{ij} t}, t \geq 0 \tag{10}$$

where the reciprocal of rates λ_{ij} and μ_{ij} are referred as *Mean Time Between Failures* (MTBF) and *Mean Time To Repair* (MTTR) of the station in industries.

As a dynamical system, the transition diagram of the above exponential reliability model is: if up, the station j in section i may go down in each infinitesimal interval δt with probability $\lambda_{ij}\delta t$; if down, it may go up during δt with probability $\mu_{ij}\delta t$. It should be point out that we focus on the *operation-dependent failure* [1] in this paper, i.e., station breakdowns cannot occur while it is in idle state. Thus, the uptime discussed above does not contain the idle time.

C. Mathematical Description of Sections

For section i, $\forall i \in \mathcal{I}$, it has three states at time t, *up*, *down* and *idle*, which is denoted as $S_i(t)$. The section is *up*, if it is working on at least one vehicle at time t, denoted as $S_i(t) = 1$; the section is *down*, if at least one of its stations is in failure at time t, denoted as $S_i(t) = -1$; otherwise the section is *idle*, denoted as $S_i(t) = 0$. The state of section i is determined by the states of its stations as follows:

$$S_i(t) = \begin{cases} 1, & \forall j \in \mathcal{J}_i, s_{ij}(t) \geq 0; \exists j' \in \mathcal{J}_i, s_{ij'}(t) = 1 \\ -1, & \exists j \in \mathcal{J}_i, s_{ij}(t) = -1 \\ 0, & \text{otherwise} \end{cases}. \tag{11}$$

The state of section buffer i is depicted by its inventory level at time t, which is denoted as $b_i(t)$. We have

$$b_i(t) = Ca_i(t) - Cd_i(t), \tag{12}$$

where $Ca_i(t)$ denotes the accumulated number of vehicles arrived at section buffer i before time t; $Cd_i(t)$ indicates the accumulated number of vehicles departed from section buffer i before time t. Obviously, $Cd_i(t) \leq Ca_i(t)$.

Section i is in idle state due to *starvation* or *blockage* of vehicles from upstream or downstream, or *starvation* of components from merging section i'. In other words, section i is idle if the input buffer is empty, $b_{i-1}(t) = 0$, or the output buffer is full, $b_i(t) = N_i$, or the merging section buffer is empty, $b_{i'}(t) = 0$.

D. Mathematical Description of Drivers

The state of driver d at time t is denoted by $y_d(t)$, $d \in \mathcal{D}$, in the following sense: $y_d(t) = 0$ if driver d is idle at central docking area at time t; $y_d(t) = l$, $l \in \mathcal{L}(d)$, if driver d is on the trip to supply lineside buffer l at time t. The MH strategy π determines the action for each driver at any time instance when the driver is idle, i.e.,

$$a_d(t) = \begin{cases} \pi_d(t, Lb_l(t), \forall l \in \mathcal{L}(d)), & \text{if } y_d(t) = 0 \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where $\pi_d(t)$ is a mapping from the lineside buffer inventories to the driver action; $a_d(t) \in \mathcal{L}(d) \cup \{0\}$. Note that in each trip, a driver can supply only one lineside buffer and the delivery package size is predetermined. The action $a_d(t)$ for driver d at time t is defined as: $a_d(t) = l, l \in \mathcal{L}(d)$, if the driver decides to supply lineside buffer l at time t; $a_d(t) = 0$, if the driver decides to take no action at time t.

E. Performance Measures

Based on the above mathematical descriptions for the stations and sections, we want to evaluate the following performance measures through simulations.

1) *Throughput*. The average number of vehicles produced by the GA line per time unit in the steady-state of system operation, which can be defined as the average arrival number of vehicles at section buffer I, i.e., the finished goods buffer at the end of the line [1]

$$TP = \lim_{T \to \infty} \frac{1}{T} \cdot Ca_I(T). \tag{14}$$

2) *Work-in process inventory* of section buffer i. The average number of vehicles contained in section buffer i of the GA line in the steady-state of its operation. It can be defined as

$$WIP_i = \lim_{T \to \infty} \frac{1}{T} \int_{t=0}^{T} b_i(t)dt, \quad \forall i \in \mathcal{I}. \tag{15}$$

3) *Parts inventory* of lineside buffer l. The average number of parts contained in lineside buffer l of the GA line in the steady-state of its operation. It can be defined as

$$H_i = \lim_{T \to \infty} \frac{1}{T} \int_{t=0}^{T} Lb_l(t)dt, \quad \forall l \in \mathcal{L} \tag{16}$$

where $Lb_l(t)$ denotes the inventory level of lineside buffer l at time t.

4) *Utilization* of driver d. The average ratio of working time over the total time for driver d in steady states. It can be defined as

$$U_d = \lim_{T \to \infty} \frac{1}{T} \sum_{u=1}^{Cs_d(T)} [Te_d(u) - To_d(u)], \quad \forall d \in \mathcal{D} \tag{17}$$

where $Te_d(u)$ indicates the time driver d ends the uth trip; $To_d(u)$ denotes the time driver d is sent out for the uth trip; $Cs_d(T)$ is the accumulated number of trips of driver d before time T.

As we mentioned in the introduction, there is a need to evaluate the above system performances in design and analysis of the GA line with MH systems. However, they could not be evaluated through closed form equations due to stochastic station reliabilities, complicated system dynamics, and various MH strategies. Thus, simulation is almost the only way to obtain the system performances under different designs.

To do efficient simulation is not an easy task. Traditional simulation methods, such as event-scheduling method, are not efficient to satisfy the requirement in practice due to large problem scales. The practical system of concern consists of multiple sections, hundreds of stations, hundreds of lineside buffers and dozens of drivers. Each item is related to several different types of events, which makes the total number of events extraordinarily large. Furthermore, the state-space of the whole system could be so large that it is impossible to be described by traditional simulation approaches. For example, each station has three states, up, down or idle. Combined together, the state-space for a section with 20 stations will reach $3^{20} \approx 10^9$. Thus, it is desirable to develop a new method, i.e., the aggregated event-scheduling method, in this paper.

IV. Aggregated Event-Scheduling Method

This section presents a novel aggregated event-scheduling method to address the challenges of efficient simulation of the GA lines with MH.

A. Two-Level Framework of Aggregated Event-Scheduling

Discrete-event simulation concerns the modeling of a system evolving over time by a representation in which system states change instantaneously at separate time points [10], [12]. These time points denote the time when events happen, where an *event* is defined as an instantaneous occurrence that may change the system state. The purpose of discrete-event simulation is to find these time points for all the events with a given random sequence and initial states, i.e., to depict a simulation trajectory (sample path) of the system within given simulation time. For the GA line with MH system, a simulation *trajectory* is defined as follows:

$$\{\xi, S_i(t), b_i(t), s_{ij}(t), \forall i \in \mathcal{I}, j \in \mathcal{J}_i, 0 \le t \le T\} \tag{18}$$

where ξ denotes a random sequence; T is the total simulation time; $S_i(0)$, $s_{ij}(0)$, $b_i(0)$ are components of system state[2] (called state variables) with initial state variables $S_i(0)$, $s_{ij}(0)$, $b_i(0)$ being given. With a simulation trajectory sufficient long, we can evaluate system performances shown in (14)–(17).

Considering the challenge of large problem scale, it is not efficient to apply traditional event-scheduling method directly to simulate the GA line with MH system since it requires maintaining a comprehensive event list, the updates of which may depend on all state variables of the entire system. To address this difficulty, the paper employs the idea of divide-and-conquer and develops a novel aggregated event scheduling method. Although general parallel simulation cannot be applied directly due to the global correlations, there are still possibilities to decouple the system based on partial decomposability. Here, the *decomposability* means that with a simulation technique, a complex system can be divided into several connected subsystems; under certain conditions, each subsystem can be simulated independently with only limited data exchange with others. The GA line with MH illustrated in Fig. 2 consists of connected sections, and yet they are separable for the purpose of simulation.

[2]It should be clear that as in GSMP model, strictly speaking, these state variables are only *discrete* state variables and the lifetime of events are also state variables if we fully determine the state transition map.

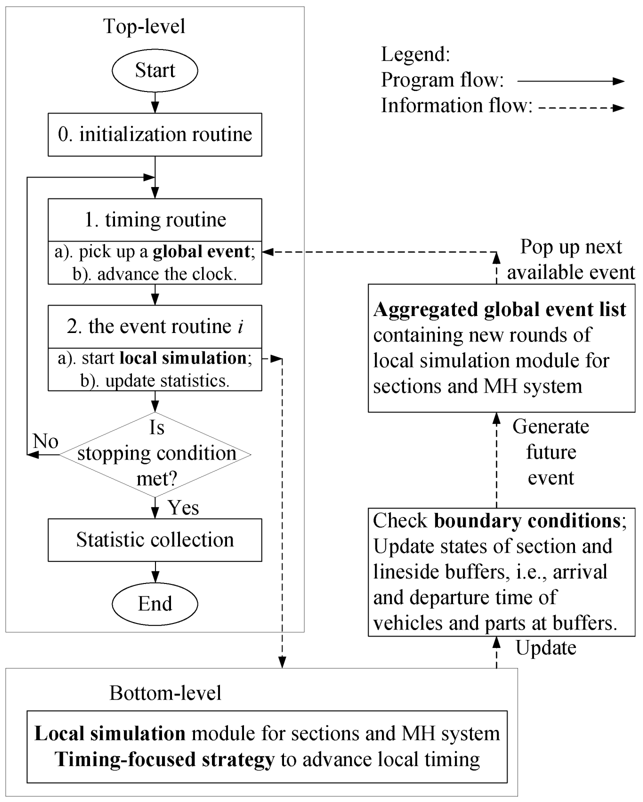

Top-level
Start
0. initialization routine
1. timing routine
a). pick up a **global event**;
b). advance the clock.
2. the event routine *i*
a). start **local simulation**;
b). update statistics.
Is stopping condition met?
No
Yes
Statistic collection
End
Legend:
Program flow:
Information flow:
Pop up next available event
Aggregated global event list containing new rounds of local simulation module for sections and MH system
Generate future event
Check **boundary conditions**; Update states of section and lineside buffers, i.e., arrival and departure time of vehicles and parts at buffers.
Update
Bottom-level
Local simulation module for sections and MH system
Timing-focused strategy to advance local timing

Fig. 3. Framework of the aggregated event scheduling method.

The aggregated event-scheduling method has a two-level framework as Fig. 3 illustrates.

The top-level simulation follows the same flow as the traditional event scheduling method and it consists of three subroutines: 1) an initialization routine which initializes the simulation clock, system states, statistical counters and event list; 2) a timing routine which determines the next event to happen and advances the simulation clock; and 3) an event routine which updates system states. The difference is: the concept of a traditional event has been extended to a "round" of local simulation for a section. To be more specific, the top level introduces a concept of *round* and divides the simulation of each section into several rounds of local simulations. Traditional events are aggregated as rounds of local simulation, each of which can be carried out separately and independently under certain boundary conditions (details will be demonstrated in Section IV-B). Therefore, the timing routine here determines the occurrences of next round of local simulation; the event routine invokes a round of local simulation to update the system state. This cycle is repeated until the stopping condition is eventually satisfied.

On the bottom-level, one round of local simulation for a section or MH system is carried out as an independent module without interactions with other parts of the system. States of section buffers are updated in bottom-level simulation, and the boundary conditions introduced in Section IV-B determine when to stop local simulation and return to the top-level. Local event lists in bottom-level are further reduced with a timing-focused simulation strategy (the details will be explained in Section IV-C), so that simulation efficiency is further improved.

To summarize, the aggregated event-scheduling method takes advantage of the divide-and-conquer idea and divides the simulation framework hierarchically into two levels based on the system partial decomposability. It has benefits as follows.

1) By dividing the simulation for each section into rounds and extending the concept of events as rounds of local simulation, the top-level of the framework keeps the size of the aggregated-event list in a reasonable scale so that the existing event scheduling concept can be extended to simulate large scale systems. A boundary condition is introduced to guarantee the equivalence of the aggregated simulation to the original problem.
2) Local event lists are further reduced with a timing-focused simulation strategy on the bottom-level, which derives timing relations of key events based on max-plus algebra and reduces redundant events.
3) The hierarchical architecture makes the simulation programming more easily and provides potentials to do simulations on parallel or distributed computers, although it is currently implemented on a serial computer.

With this new method it is possible to mimic real production problems fast and accurately within a reasonable computational time frame. Details of these features will be explained in the remaining part of this section.

B. Top-Level: Dividing Mechanism and Boundary Conditions

As the previous subsection explained, the purpose of the simulation for section i is to get its trajectory, which consists of M_i conveyor moves within the given simulation time T. Thus, the key of the simulation is to get the time of the mth conveyor-move in section i, which is denoted as $Tc_i(m)$, $m = 1, 2, \ldots, M_i$. Here, we are facing a challenge to do efficient simulation since we cannot get all of the conveyor moves at the same time due to the correlations between different sections. However, the partial decomposability of the system provides a way to do local simulation of a section separately and independently within a short time-window, and obtain starting times of several conveyor moves. In other words, we need to introduce boundary conditions based on the states of section buffers and divide the top-level simulation of each section into rounds, so that simulation can be done alternatively among sections. Even though production of a section continues without stopping in the real situation, we have to stop a round of local simulation for the section if boundary conditions are not satisfied.

Notation: In the top-level dividing mechanism, we introduce the concept of *round* and divide the simulation of section i into R_i rounds; the rth round contains m_{ir} conveyor moves, which can be simulated separately and independently. First, we introduce the following notation to explain this mechanism.

M_i	Total number of conveyor moves in section i within the given total simulation time T.
R_i	Total number of local simulation *rounds* of section i.
m_{ir}	Number of conveyor moves during the r round of local simulation of section i; we have $M_i = \sum_{r=1}^{R_i} m_{ir}$.
t_{ir}	Starting time of the r round local simulation of section i.
e_{ir}	Ending time of the rth round local simulation of section i. e_{ir} can be determined by t_{ir} and m_{ir} during the rth round local simulation of section i.

1) Boundary Conditions: Following similar ideas in our previous work [37], boundary conditions for the dividing mechanism in top-level simulation contain the following three parts. Suppose section i has finished r_i rounds of local simulation. The condition and the time to start the $r_i + 1$th round local simulation and the number of conveyor moves in the $r_i + 1$th round local simulation are determined as follows.

2) (P1): The condition to start the $r_i + 1$th round local simulation for serial section i, $i \in \mathcal{I}_s$, is determined by the following (19)–(22):

$$Cm_{i1}(e_{ir_i}) + 1 \le Ca_{i-1}(e_{i-1,r_{i-1}}) \tag{19}$$

$$Cm_{iJ_i}(e_{ir_i}) - N_i \le Cd_i(e_{i+1,r_{i+1}}) \tag{20}$$

$$Cm_{iE_i}(e_{ir_i}) + 1 \le Ca_{i'}(e_{i'r'_i}) \tag{21}$$

$$Cu_l(e_{ir_i}) + P_l(Cm_{ij}(e_{ir_i}) + 1) \le Cr_l(e_{ir_i}), \ \forall j \in \mathcal{J}_i, l \in \mathcal{L}_{ij} \tag{22}$$

where the notation has the following meanings:

$Cm_{ij}(t)$	Accumulated number of vehicles processed on station j in section i before time t.
E_i	Index of the station where merging section i' join into the serial section i. Here, we assume that each serial section has one merging section at most.
$Cu_l(t)$	Accumulated number of parts used at lineside buffer l before time t.
$P_l(k)$	Number of parts used by the kth vehicle at lineside buffer l.
$Cr_l(t)$	Accumulated number of parts replenished at lineside buffer l before time t.

Hereafter, we let $N_I = \infty$ and $Ca_0(t) = \infty$ for all t according to assumption (5) in Section III-A.

The physical meaning of these conditions is intuitive: (19) and (20) guarantee we have sufficient information about the upstream and downstream section buffers of serial section i, i.e., its input buffer is not empty and output buffer is not full; (21) and (22) make sure that the component from the merging section and the parts from all lineside buffers are ready before we carry out the $r_i + 1$th round local simulation for serial section i.

The condition to start the $r_i + 1$th round local simulation for merging section i, $i \in \mathcal{I}_m$, is determined by (20)(22), since materials at the input buffers of merging sections are always available and there are no further merges in the merging section in the current system we are concerned.

3) (P2): The starting time of the $r_i + 1$th round local simulation for serial section i, $i \in \mathcal{I}_s$, is determined as follows: If $s_{ij}(e_{ir_i}) = 0, \forall j \in \mathcal{J}_i$, then

$$t_{i,r_i+1} = \max\{e_{ir_i}, Ta_{i-1}(Cm_{i1}(e_{ir_i}) + 1)\};$$

otherwise, see equation (23) at the bottom of the next page, where the notation has the following meanings.

$Ta_i(k)$	Arrival time of vehicle k at section buffer i.
$Td_i(k)$	Departure time of vehicle k at section buffer i. We set $Td_i(k) = 0$ if $k \leq 0$.
$Tr_l(p)$	Replenishment time of part p at lineside buffer l.

The physical meaning of (*P2*) is intuitively clear. There are two cases when we determine the starting time of the $r_i + 1$th round local simulation of the serial section i. If all stations in section i are empty after r_i rounds of local simulation, i.e., $s_{ij}(e_{ir_i}) = 0, \forall j \in \mathcal{J}_i$, the starting time is determined by the finishing time of r_i rounds of simulation and the arrival time of vehicle $Cm_{i1}(e_{ir}) + 1$ at section buffer $i - 1$. Otherwise, some station in section i is not empty, then the starting time of next round of local simulation is determined by the finishing time of r_i rounds of the simulation, departure time of vehicle $Cm_{iJ_i}(e_{ir}) - N_i$ at section buffer i, ready time of the next component for station E_i from merging section i', and arrival time of needed parts at all lineside buffers of serial section i.

Similarly, the starting time of the $r_i + 1$th round local simulation for merging section i, $i \in \mathcal{I}_m$, is determined by (24) at the bottom of the page.

4) (P3): The number of conveyor moves in the $r_i + 1$th round local simulation of serial section $i, i \in \mathcal{I}_s$, is determined by (25). The intuition behind (25) is: the $r_i + 1$th round local simulation of serial section i has to stop once information of input or output buffers, or the merging section, or any lineside buffer of serial section i is not available

$$m_{i,r_i+1} = \min \left\{ \begin{array}{c} Ca_{i-1}(e_{i-1,r_{i-1}}) - Cm_{i1}(e_{ir_i}), \\ Cd_i(e_{i+1,r_{i+1}}) + N_i - Cm_{iJ_i}(e_{ir_i}), \\ Ca_{i'}(e_{ir_i}) - Cm_{iE_i}(e_{ir_i}), \\ \min_{\forall j \in \mathcal{J}_i, l \in \mathcal{L}_{ij}} \{K_{ijl}(e_{ir_i})\} \end{array} \right\} \quad (25)$$

where

$$K_{ijl}(t) := \max \left\{ K : Cr_l(t) \geq Cu_l(t) + \sum_{k=1}^{K} P_l(Cm_{ij}(t) + k) \right\} \quad (26)$$

which is the remaining lifetime of lineside buffer l at time t, i.e., how many steps of conveyor-moving it can maintain without replenishment. Similarly, the number of conveyor moves in the $r_i + 1$th round local simulation of merging section i, $i \in \mathcal{I}_m$, is determined by

$$m_{i,r_i+1} = \min \left\{ \begin{array}{c} Cd_i(e_{i+1,r_{i+1}}) + N_i - Cm_{iJ_i}(e_{ir_i}), \\ \min_{\forall j \in \mathcal{J}_i, l \in \mathcal{L}_{ij}} \{K_{ijl}(e_{ir_i})\} \end{array} \right\} \quad (27)$$

(P1)–(P3) outline the boundary conditions for the dividing mechanism on top-level simulation, which plays as the guidance of the aggregated event-scheduling method. With the arrival and the departure time of vehicles at section buffers, and the replenishment time of parts at lineside buffers, we could obtain the time to start next round local simulation in (24) and the number of conveyor moves in each round local simulation in(25) and (27), which play the similar role as null messages used in [39], [42], and [43]. This dividing mechanism improves the simulation efficiency significantly due to the following reason: with the condition in (*P1*) and the time given by *(P2)*, we can run a round of local simulation for a section with smaller scales of system states and event list, without communication with other sections, for a given number of conveyor moves predetermined by (*P3*).

5) Validations for the Boundary Conditions: This subsection validates the correctness of the boundary conditions of the dividing mechanism on top-level with the following property. The *correctness* here means that there is not deadlock according to the boundary conditions.

6) Property 1: There is no deadlock according to the boundary conditions of the top-level dividing mechanism. In other words, at any time there exists section $i, i \in \mathcal{I}$, satisfying conditions in (*P1*) to start another round of local simulation.

Proof: No deadlock would happen for the GA line without MH since there is no cycle in the line. Even though deadlock may happen when simulating the entire system with GA line and MH, it is avoidable due to the following reasons. The boundary conditions (19)–(27) in the top level dividing mechanism determine the time to start the next round local simulation and the number of conveyor moves in each round. These time and numbers of all sections are distributed in the top level simulation, similar to the null messages used in [39], [42], and [43], so that the deadlock is avoidable here with the same reason as null message methods in [39], [42], and [43]. ■

$$t_{i,r_i+1} = \max \left\{ Td_i(Cm_{iJ_i}(e_{ir_i}) - N_i), Ta_{i'}(Cm_{iE_i}(e_{ir_i}) + 1), e_{ir_i}, \max_{\forall j \in J_i, l \in L_{ij}} \{Tr_l(Cu_l(e_{ir_i}) + P_l(Cm_{ij}(e_{ir_i}) + 1))\} \right\} \quad (23)$$

$$t_{i,r_i+1} = \max \left\{ e_{ir_i}, Td_i(Cm_{iJ_i}(e_{ir_i}) - N_i), \max_{\forall j \in \mathcal{J}_i, l \in \mathcal{L}_{ij}} \{Tr_l(Cu_l(e_{ir_i}) + P_l(Cm_{ij}(e_{ir_i}) + 1))\} \right\} \quad (24)$$

The above property provides theoretical validations to the boundary conditions in the top-level dividing mechanism. Numerical validations will be provided in Section V.

7) Computation Complexity Analysis on Top-Level: This section analyzes the computation complexity analysis of the top-level dividing mechanism with boundary conditions. The *computation complexity* here indicates the time complexity of the simulation method, i.e., the number of steps the method takes to simulate a given system as a function of the size of the system.

First, we estimate the computation budgets of the traditional event-scheduling method, if it is directly used to solve the problem. By *computation budget*, we mean the CPU time needed to simulate a given system as a function of the system size. For station j in section i, there are six types of events: starts or finishes an operation, fails or repairs, loads or unloads of vehicles. Let J denote the total number of stations in the production line, we have

$$J = \sum_{i \in \mathcal{I}} J_i. \tag{28}$$

The average length of the event list will be about $3J$. In traditional event scheduling methods, the major computation task is to sort over the event list, which is needed every time an event happens. In the GA line with MH system, the production rate is so fast that conveyors will move in every minute on average. With each conveyor move, there will be four events occurring at each station. Therefore, the frequency of events in the system is very high, about $4J$ events every minute on average. Taking a practical system as an example with total simulation time as one week (about 10^4 minutes), we have $J = 144$, which implies that it is needed to do about $10^4 * 4J = 5 * 10^6$ sorting operations over an event list with length about $3J = 432$ on average. A numerical test for these sorting operations is done on a PC with 2.80GHz CPU and 2.00GB RAM. The result shows that the above sorting operations will need about 32 hours, which obviously cannot meet the practical requirement.

The computation budget of sorting operation over an event list with length n is proportional to $(n \log_2 n)$. With the top-level dividing mechanism with boundary conditions, the aggregated event-scheduling method divides the whole event list into I local event lists, so that we only need to do sorting operations over aggregated event list with length n/I. Note that the total number of sorting operations is not changing. Thus, the computation budget of aggregated-event scheduling method is proportional to $(n/I \ \log_2(n/I))$, which is about $1/I$ of the traditional one. If we use an alternative simulation operation by maintaining a sorted event list and inserting new generated future events in appropriate places, the computation budget is proportional to n for inserting operation over an event list with length n. Based on the same argument as above, the computation budget of the inserting operation with the top-level dividing mechanism and boundary conditions is proportional to (n/I), which is also $1/I$ of the traditional one. Therefore, as long as we use the same operation (either inserting or sorting) in both the new simulation method and the traditional one, we could save $1/I$ computation budget with the top-level dividing mechanism. This analysis demonstrates the efficiency of the new method. Even though it is a limitation that the improvement of the new method depends on the number of sections, this result is still promising when dealing with large scale practical systems typically with plenty of sections.

C. Bottom-Level: Timing-Focused Simulation Strategy

With the dividing mechanism with boundary conditions on top-level, simulations for sections are divided into rounds of local simulations, and they are carried out here on the bottom-level. To further improve the efficiency, local event lists are further reduced on the bottom-level with a timing-focused simulation strategy, by focusing on the time points of key events and deriving relations of other events with max-plus algebra, the details of which is explained in the online technique report [50].

In the local simulation for sections, conveyor moves are the key events to focus. The time of a conveyor move is determined by the following three aspects: 1) the time when all stations in the section finish their current operations; 2) the time when the output buffer of the section gets ready to store the next vehicle; and 3) the merging buffer of the section has content in it. Based on the time of conveyor moves, other events, such as the arrival and departure of vehicles at section buffers, the start and finish of assembly processes in stations, can be determined according the dynamic of the system. This local simulation for sections here is similar to the simulation of serial lines shown in our previous work [37].

In the local simulation for the MH system, the start of supplying trips are the key events to focus. During the assembly process, lineside buffers generate requests for parts to certain drivers. The driver serves the requests based on the first come first serve (FCFS) principle, which can be simulated similarly to the queueing system as [40], by considering the drivers as servers and the requests as the customers.

The timing-focused simulation strategy is efficient since it takes advantage of the specific structures of this problem and only focuses on the time of key event, instead of time of all events. Based on these key timings, other timings related to the production process can be inferred through simple formulas, i.e., the entire trajectory is accurately regenerated. Therefore, as shown in [50], this strategy presents an elegant performance and improves the simulation efficiency significantly.

D. Discussion on the Simulation Method

As we have pointed out in the literature review, the general principles for parallel and distributed discrete-event simulation has provided us useful insights in designing the top-level simulation; however, nontrivial work has to be done to harvest benefits of parallel and distributed simulation for our entire system. Below are the major differences of our work compared with existing work from three aspects.

1) The FCFS queueing network model studied in [40] does not provide an efficient way to decompose our system. The reason is the dynamic of sections in the GA line with MH is much more complicated and different from the queueing network studied in literatures. Specifically, the state transition of each station relies on the status of all other stations within one section since they are strongly coupled by the conveyor; whereas individual stations are

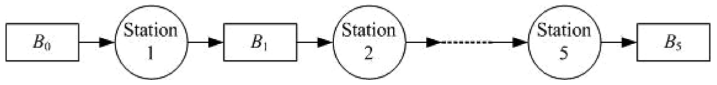


Fig. 4. Layout of five-station serial line.

different since they have different MTBF, MTTR, and various consumption rates for parts. Even worse, if with the queueing network formulation, each station would have one queue for semi-produced vehicles and queues for parts. Since there are more than 100 stations and more than 300 lineside buffers in our system, the size of the queueing network would be huge, which makes the traditional method inefficient.

2) The appointment protocol with lookahead proposed in [40] and [42] cannot be directly used for our problem since otherwise we might end up with limited performance improvement due to the following reasons: (a) If we use the lookahead mentioned in [42], the lookahead time might be short since the minimum increment of a *Logical Process* (LP) for processing any event is very small in our problem, (in other words, the cycle time of the assembly is very small, about 1 minute). If we use the scheme proposed in [40], the lookahead mostly would equal to the cycle time due to the serial structure of the GA line. Thus, the speedup with such short lookahead is limited as it is pointed out in [40] that: the ability to "reduce synchronization overhead to acceptable levels clearly depends on the ability of provide lookahead." (b) The service time in our system (i.e., the assembly time for semi-produced vehicles on each station) is with high variation (each station has random failures and repair time; all stations in one section are highly coupled with one conveyor). As it is pointed out in [40], "under high variation of service time, very small lookahead values are possible, meaning that lookahead is computed more often, thereby incurring increased overhead."

3) Although we use the basic idea of asynchronous protocols in distributed simulation, it has to be extended to fit our problem. Instead of buffer level (queue length), we record the arrival and departure time of semi-produced vehicles at section buffers. With the information of vehicle arrival and departure time, we cannot only regenerate the buffer level according to (12), but also decompose the entire simulation into local simulation for sections as the top-level dividing mechanism shows, rather than the message channel [39], [42] and the lookahead method [40], [43]. Instead of the appointment protocol [40], we provide the condition and time to start and stop local simulation of each section (i.e., Logical processes) in (19)–(27), based on the information of vehicle arrival and departure time recorded at section buffers. Instead of small event lists in logical processes, we use max-plus algebra to capture the dynamics of the strongly coupled stations in each section in the local simulation.

To summarize entire Section IV, the aggregated event scheduling simulation method introduces a hierarchical architecture into simulations and designs a two-level framework to deal with the challenge of large problem scales. Based on the specific structure of the GA line with MHs, a dividing mechanism with boundary conditions is employed on the top level simulation, so that events are aggregated as rounds of local simulations and the entire event list is divided into pieces. Additionally, a timing-focuses strategy based on max-plus algebra is applied in rounds of local simulation on bottom-level, so that local event lists are further reduced. The effectiveness and efficiency of this aggregated event-scheduling method are demonstrated theoretically through properties and computation complexity analysis. Numerical experiments will be shown in the next section.

V. Numerical Experiments and Insights

This section provides two experiments to demonstrate the effectiveness and the efficiency of the aggregated event-scheduling method. All the tests are implemented with programming language C++ and tested on a PC with Pentium D 2.80 GHz CPU, 2.00 GB RAM, Windows system.

A. Effectiveness and Efficiency of the Simulation Method

For comparison purposes, the aggregated event-scheduling method and the traditional one are both implemented on a five-station assembly line shown in Fig. 4 without MH.

We develop both the traditional event scheduling scheme [10], [12] and the aggregated event-scheduling method in C++ program to simulate the system above. Additionally, an Arena simulation is carried out for the same systems. We use 32 groups of test cases with different parameters, the detailed parameters of which are listed in the online report [50].

We compare the simulation results and CPU time of the aggregated event-scheduling method, traditional event scheduling scheme, and Arena software with these 32 test cases. The simulation results for system throughput of these three different simulation approaches are shown in Table I. Columns 2, 3, and 4 are the simulated throughput in Jobs/hour of 32 test cases with these three approaches respectively. Columns 5 and 6 are the relative differences between the aggregated event scheduling method versus Arena and traditional scheme. The last two rows of Table I show the maximum and the minimum of the absolute values of the relative differences comparing the new method with Arena and traditional event scheduling scheme.

Table I illustrates the effectiveness of the aggregated event scheduling method, since it provides almost the same (less than 1% relative difference) simulation results as the traditional event scheduling scheme and software Arena, with only 0.01% and 0.22% relative differences on average, respectively. For each test case, we do simulation for 20 replications with both the aggregated and the traditional event scheduling methods. The average simulation results with these two methods are shown in columns 3 and 4 of Table I, respectively. We perform a t-test of the null hypothesis that the difference between simulated throughput with these two methods are a random sample from a normal distribution with mean 0 and unknown variance, against the alternative that the mean is not 0. All 32 test cases return

TABLE I
EFFECTIVENESS OF AGGREGATED EVENT-SCHEDULING: COMPARISON OF SIMULATION RESULTS OF THREE APPROACHES

Case #	Arena result	Traditional event scheduling C++	Aggregated event scheduling C++	Diff. with Arena (%)	Diff. with tradi. (%)	p-value
1	46.21	46.02	46.05	0.34	-0.06	0.49
2	49.78	49.78	49.75	0.07	0.06	0.38
3	45.87	45.81	45.82	0.11	-0.02	0.86
4	49.68	49.78	49.72	-0.08	0.12	0.41
5	35.29	35.27	35.21	0.24	0.17	0.07
6	39.17	39.26	39.19	-0.05	0.18	0.06
7	35.05	34.74	34.75	0.86	-0.02	0.89
8	39.18	39.12	39.05	0.33	0.18	0.27
9	35.27	35.21	35.20	0.20	0.02	0.86
10	39.18	39.22	39.20	-0.05	0.04	0.68
11	34.98	34.64	34.74	0.70	-0.29	0.15
12	39.10	39.00	39.10	0.01	-0.25	0.27
13	37.65	37.51	37.53	0.32	-0.05	0.67
14	40.81	40.83	40.86	-0.12	-0.07	0.48
15	37.34	37.21	37.18	0.42	0.06	0.81
16	40.74	40.73	40.72	0.05	0.02	0.88
17	36.31	36.19	36.16	0.42	0.10	0.44
18	39.85	39.86	39.85	-0.01	0.00	0.98
19	35.91	35.60	35.69	0.62	-0.24	0.20
20	39.79	39.71	39.65	0.35	0.15	0.55
21	34.43	34.37	34.40	0.09	-0.08	0.41
22	38.43	38.38	38.40	0.08	-0.05	0.67
23	34.02	33.87	33.80	0.65	0.22	0.32
24	38.28	38.29	38.27	0.03	0.05	0.78
25	35.85	35.81	35.84	0.04	-0.07	0.47
26	39.28	39.33	39.31	-0.07	0.06	0.49
27	35.55	35.42	35.38	0.48	0.12	0.57
28	39.21	39.20	39.10	0.29	0.27	0.18
29	35.59	35.52	35.54	0.14	-0.05	0.72
30	39.91	39.86	39.86	0.13	0.00	0.98
31	35.19	35.01	35.04	0.43	-0.10	0.64
32	39.86	39.82	39.82	0.10	-0.01	0.95
Ave				0.22	0.01	
Max				0.86	0.29	
Min				0.01	0.00	

$h = 0$, i.e., failures to reject the null hypothesis at the 1% significance level. The p-values are shown in column 7 of Table I. This experiment shows that the aggregated event scheduling method provides the same results as the traditional method statistically.

Table II demonstrates the efficiency of the aggregated event scheduling method. Given the almost identical simulation results, these three approaches need various CPU time. Table II compares the average CPU time (for each replication) over the 32 groups of test cases. The total simulation time is 100,000 minutes, and no animation is included in Arena to make the comparison fair. From Table II, we can clearly see that aggregated event scheduling needs much less CPU time than software Arena and traditional event scheduling need. Arena needs long CPU time due to the difference of platform and the total simulation time (the setting with 100,000 minutes total simulation time is much longer than usual settings in Arena). With the same C++ implementation platform, the aggregated event scheduling method on average saves about 52 times computation budget comparing with the traditional one, which illustrates the tremendous improvement on simulation efficiency of the new method.

B. Validation With Production Data From a Practical System

We implement the aggregated event-scheduling simulation method on a real-world GA line with MH systems in automotive industry. The scale of the practical system is with several sections (including trim, chassis, door, final, etc.), more than 150 stations, more than 300 lineside buffers, and dozens of drivers. For confidential considerations, the parameters of this practical system cannot be released in public. The warm-up time, the total simulation time and the replication number are set to 5000 minutes, 10,000 minutes, and 20, respectively, according to the following statistic tests shown in Tables III and IV. Table III shows the t-tests for two simulation setups: one is with 5000 minutes warm-up time and 10,000 minutes total simulation time; the other is with 10,000 minutes warm-up time and 15,000 minutes total simulation time. Both setups are with 20 replications. We perform t-tests for the throughput (*TP*) and individual utilizations of 16 drivers (U_d, $d = 1, \ldots, 16$) obtained from these two simulation setups. Note that simulation results are modified for the business confidential considerations, but the modification does not add any inaccuracy in this comparison. (The same as Table IV does.) The null hypothesis of each test is that the differences of simulation results between these two setups are random samples from a normal distribution with mean 0, against the alternative that mean is not 0. All tests return $h = 0$, i.e., failures to reject the null hypothesis at the 1% significance level, with the p-values shown in columns 4 and 8 in Table III. This result illustrates that the setup with 5000 minutes warm-up time and 10,000 minutes total simulation time is statistically sufficient.

Similarly to Table III, Table IV shows the t-tests for another two simulation setups: one is with 20 replications and the other is with 50 replications. Both of them are with 5000 minutes warm-up time and 10,000 minutes total simulation time. The null hypothesis of each test is that the differences of simulation results between these two setups are random samples from a normal distribution with mean 0, against the alternative that mean is not 0. All t-tests return $h = 0$, i.e., failures to reject the null hypothesis at the 1% significance level, with the p-values shown in columns 4 and 8 in Table IV. This result demonstrates that the setup with 20 replications is statistically sufficient.

TABLE II
EFFICIENCY OF AGGREGATED EVENT-SCHEDULING:COMPARISON OF CPU TIME OF THREE APPROACHES

Approach	Arena	Traditional event scheduling in C++(a)	Aggregated event scheduling in C++(b)	Time saving vs traditional (a)/(b)
CPU time	$\approx$ 42s	8.098s	0.155s	52.24

TABLE III
STATISTICAL TESTS FOR DIFFERENT WARM-UP TIME

Measure	5,000 min. Warm-up time	10,000 min. Warm-up time	p-value	Measure	5,000 min. Warm-up time	10,000 min. Warm-up time	p-value
TP	29.79	29.75	0.27	U_9	0.62	0.62	0.43
U_1	0.53	0.53	0.47	U_{10}	0.80	0.80	0.10
U_2	0.51	0.50	0.03	U_{11}	0.69	0.69	0.13
U_4	0.69	0.69	0.14	U_{12}	0.69	0.68	0.38
U_4	0.61	0.61	0.53	U_{13}	0.58	0.58	0.69
U_5	0.65	0.65	0.26	U_{14}	0.56	0.56	0.53
U_6	0.68	0.67	0.05	U_{15}	0.51	0.51	0.81
U_7	0.63	0.63	0.73	U_{16}	0.43	0.43	0.51
U_8	0.68	0.68	0.93				

TABLE IV
STATISTICAL TESTS FOR DIFFERENT REPLICATION NUMBERS

Measure	Replication #=20	Replication #=50	p-value	Measure	Replication #=20	Replication #=50	p-value
TP	29.79	29.80	0.69	U_9	0.62	0.62	0.69
U_1	0.53	0.53	0.82	U_{10}	0.80	0.80	0.29
U_2	0.51	0.51	0.61	U_{11}	0.69	0.69	0.92
U_3	0.69	0.69	0.93	U_{12}	0.69	0.69	0.70
U_4	0.61	0.61	0.59	U_{13}	0.58	0.58	0.97
U_5	0.65	0.65	0.58	U_{14}	0.56	0.56	0.39
U_6	0.68	0.68	0.59	U_{15}	0.51	0.51	0.88
U_7	0.63	0.63	0.35	U_{16}	0.43	0.43	0.61
U_8	0.68	0.68	0.54				

TABLE V
DIFFERENCES OF SIMULATED AND MEASURED UTILIZATIONS

Driver index	Average Difference(%)	Confidence Interval with 90% Confidence level(%)	
#1	2.42	2.40	2.44
#2	2.90	2.88	2.92
#3	2.45	2.43	2.46
#4	2.21	2.21	2.22
#5	2.40	2.39	2.42
#6	2.55	2.52	2.57
#7	2.36	2.35	2.38
#8	0.02	0.00	0.04
#9	2.44	2.42	2.45
#10	2.46	2.45	2.47
#11	1.91	1.90	1.92
#12	2.44	2.41	2.46
#13	1.27	1.26	1.28
#14	2.51	2.49	2.53
#15	2.55	2.53	2.58
#16	2.54	2.52	2.55
Mean	2.21	2.20	2.23

TABLE VI
DIFFERENCES OF SIMULATED AND MEASURED THROUGHPUT

	Average Difference(%)	Confidence Interval with 90% Confidence level(%)	
TP	2.21	2.11	2.13

We compare the simulation results with the practical production data which is measured and collected for one month in a factory. The results are shown in Tables V and VI.

Tables V and VI demonstrate the effectiveness of the aggregated event scheduling method. Table V shows that the simulated utilizations of 16 drivers are close to the measured ones (with 2.21% relative difference and [2.20%, 2.23%] confidence interval for 90% confidence level on average). Table VI shows that the simulated throughput is close to the measured one (with 2.12% relative difference and [2.11%, 2.13%] confidence interval for 90% confidence level). This discrepancy is due to some human factors which are simplified in simulation. This accuracy is good considering the fact that there is typically about 5% measure error in data collection.

Additionally, the above experiment also demonstrates the efficiency of the aggregated event-scheduling method. We should notice that the CPU time of our new simulation method is only 1.85 seconds per replication for the practical system with 10,000 minutes (about 7 days) total simulation time with good match for the measured throughput and utilization data. This simulation speed illustrates the efficiency of the new method.

The last but not the least, the simulation method developed in this paper is helpful to answer "what-if" questions and produce managerial insight on the behavior of the GA line with MH system, since this method can simulate large scale systems within acceptable computation time. The detailed of this part is shown in the online technique report [50]. Additionally, the simulation method can also be applied to do simulation-based optimization and analysis for the GA line with MH system, which are shown in our recent work [38], [41].

VI. CONCLUSION

This paper presents a novel simulation method for a GA line with MH systems to meet the challenges of large problem size and correlated system dynamics. Different from the traditional event scheduling based methods, it combines ideas of max-plus algebra, event scheduling, and decoupled simulation. Making use of the partial system decomposability, we introduce a two-level simulation framework. A dividing mechanism with boundary conditions is employed on top-level to divide the global event list into small sizes. A timing-focused strategy based on max-plus algebra is applied on bottom-level local simulation to further reduce local event lists. With this new method it is possible to mimic real production systems fast and accurately within a reasonable computational time frame. Numerical testing results of a practical production line show that the new method is efficient and effective. Additionally, although this simulation method is developed for the GA line with MH systems, it should be pointed out that this method is also applicable to simulate other manufacturing systems (such as disassembly lines), and systems with fork/join structures (such as parallel processing systems and distributed replicated database systems).

This simulation method can be applied to investigate various scenarios and analyze parameter sensitivity. More importantly, engineers and plant managers could make real time decisions via this fast and accurate model. Furthermore, simulation based optimization is being conducted systemically and will be reported in our future work.

ACKNOWLEDGMENT

The authors would like to thank Prof. Y.-C. Ho, Prof. X.-R. Cao, Prof. L. Shi, Dr. Q.-S. Jia, and T. Sun for their valuable comments and suggestions. They also wish to thank the anonymous reviewers for their insightful comments.

REFERENCES

[1] J. Li and S. M. Meerkov, *Production Systems Engineering*. New York: Springer, 2009.
[2] E. J. Williams and H. Celik, D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, Eds., "Analysis of convey or systems within automotive final assembly," in *Proc. 1998 Winter Simulation Conf.*, pp. 915–920.
[3] S. K. Bhattacharyya, R. Roy, and M. J. Low, "A computer simulation system for the evaluation of man assignments on car assembly tracks," *Simulation*, vol. 61, no. 2, pp. 124–133, 1993.

[4] Y.-L. Chang, R. S. Sullivan, and J. R. Wtlson, "Using SLAM to design the material handling system of a flexible manufacturing system," *Int. J. Prod. Res.*, vol. 24, no. 1, pp. 15–26, 1986.

[5] A.-V. Ardavan and L. Gilbert, "Loop based facility planning and material handling," *Eur. J. Oper. Res.*, vol. 164, pp. 1–11, 2005.

[6] S. B. Gershwin, *Manufacturing Systems Engineering*, S. B. Gershwin, Ed. Englewood Cliffs, N.J.: Prentice-Hall, 1994.

[7] B. M. Beamon, "Performance, reliability, and performability of material handling systems," *Int. J. Prod. Res.*, vol. 36, no. 2, pp. 377–393, 1998.

[8] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Boston, MA: Kluwer Academic, 1999.

[9] J. Li, D. E. Blumenfeld, N. Huang, and J. M. Alden, "Throughput analysis of production systems: Recent advances and future topics," *Int. J. Prod. Res.*, vol. 47, no. 14, pp. 3823–3851, 2009.

[10] J. Banks, J. S. Carson, II, B. L. Nelson, and D. M. Nicol, *Discrete-Event System Simulation*. Upper Saddle River, NJ: Prentice-Hall, 2005.

[11] J. A. Harding and K. Popplewell, "Simulation: An application of factory design process methodology," *J. Oper. Res. Soc.*, vol. 51, no. 4, pp. 440–448, 2000.

[12] A. M. Law and W. D. Kelton, *Simulation Modeling and Analysis*. New York: McGraw-Hill, 2000.

[13] J. S. Smith, "Survey on the use of simulation for manufacturing system design and operation," *J. Manuf. Syst.*, vol. 22, no. 2, pp. 157–171, 2003.

[14] G. Weigert, S. Horn, and S. Werner, "Optimization of manufacturing processes by distributed simulation," *Int. J. Prod. Res.*, vol. 44, no. 18–19, pp. 3677–3692, 2006.

[15] W. Wang and R. Bell, "A knowledge based multi-level modelling system for the design of flexible machining facilities," *Int. J. Prod. Res.*, vol. 30, no. 1, pp. 13–34, 1992.

[16] A. Grosfeld-Nir, M. Magazine, and A. Vanberkel, "Push and pull strategies for controlling multistage production systems," *Int. J. Prod. Res.*, vol. 38, no. 11, pp. 2361–2375, 2000.

[17] A. Grosfeld-Nir and M. Magazine, "A simulation study of pull systems with ascending/descending buffers and stochastic processing times," *Int. J. Prod. Res.*, vol. 43, no. 17, pp. 3529–3541, 2005.

[18] A. W. Booth, "Object-oriented modeling for flexible manufacturing systems," *Int. J. Flexible Manuf. Syst.*, vol. 10, pp. 301–314, 1998.

[19] L. Rabelo, M. Helal, A. Jones, and H.-S. Min, "Enterprise simulation: A hybrid system approach," *Int. J. Comput. Integr. Manuf.*, vol. 18, no. 6, pp. 498–508, 2005.

[20] S. Kumar and D. A. Nottestad, "Capacity design: An application using discrete-event simulation and designed experiments," *IIE Trans.*, vol. 38, pp. 729–736, 2006.

[21] R. A. Lindau, T. Kanflo, and K. R. Lumsden, "Impact of real-time information for scheduling a car-body shop – A simulation study," *Int. J. Oper. Prod. Manage.*, vol. 14, no. 3, pp. 114–125, 1994.

[22] F. T. S. Chan, "Using simulation to predict system performance: A case study of an electro-phoretic deposition plant," *Integr. Manuf. Syst.*, vol. 6, no. 5, pp. 27–38, 1995.

[23] A. Jayaraman, R. Narayanaswamy, and A. K. Gunal, S. Andrado'ttir, K. J. Healy, D. H. Withers, and B. L. Nelson, Eds., "A sortation system model," in *Proc. 1997 Winter Simulation Conf.*.

[24] G. M. Buxey and D. Sadjadi, "Simulation studies of conveyor-paced assembly lines with buffer capacity," *Int. J. Prod. Res.*, vol. 14, no. 5, pp. 607–624, 1976.

[25] A. Prakash and M. Chen, "A simulation study of flexible manufacturing systems," *Comput. Ind. Eng.*, vol. 28, no. I, pp. 191–199, 1995.

[26] S. N. Kumar and R. Sridharan, "Simulation modeling and analysis of tool sharing and part scheduling decisions in single-stage multimachine flexible manufacturing systems," *Robot. Comput.-Integr. Manuf.*, vol. 23, pp. 361–370, 2007.

[27] K. S. El-Kilany, P. Yong, and M. A. El Baradie, "Generic tool for modelling and simulation of semiconductor intrabay material handling system," *J. Mater. Process. Technol.*, vol. 155–156, pp. 1927–1934, 2004.

[28] S. H. Kong, "Two-step simulation method for automatic material handling system of semiconductor fab," *Robot. Comput.-Integr. Manuf.*, vol. 23, pp. 409–420, 2007.

[29] Y. Han, D. Park, S. Chae, and C. Lee, "Full fabrication simulation of 300 mm wafer focused on AMHS (automated material handling systems)," *Lecture Notes In Computer Science*, vol. 3398, pp. 514–520, 2005.

[30] X.-R. Cao and Y.-C. Ho, "Models of discrete event dynamic systems," *IEEE Control Syst. Mag.*, vol. 10, no. 4, pp. 69–76, 1990.

[31] D.-Z. Zheng and Q. Zhao, *Discrete Event Dynamic Systems* (in Chinese). Beijing, China: Tsinghua Univ. Press, 2001.

[32] G. Cohen, D. Dubois, J. P. Quadrat, and M. Viot, "A linear-system theoretic view of discrete-event processes and its use for performance evaluation in manufacturing," *IEEE Trans. Autom. Control*, vol. 30, pp. 210–220, 1985.

[33] S. R. Das, "Adaptive protocols for parallel discrete event simulation," *J. Oper. Res. Soc.*, vol. 51, no. 4, pp. 385–394, 2000.

[34] R. M. Fujimoto, B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, Eds., "Parallel and distributed simulation systems," in *Proc. 2001 Winter Simulation Conf.*, pp. 147–157.

[35] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Belmont, MA: Athena Scientific, 1997.

[36] R. R. Inman, "Empirical evaluation of exponential and independence assumptions in queuing models of manufacturing systems," *Prod. Oper. Manage.*, vol. 8, no. 4, pp. 409–432, 1999.

[37] Y. Zhao, C.-B. Yan, Q. Zhao, N. Huang, J. Li, and X. Guan, "Efficient simulation for serial production lines based on aggregated event-scheduling," in *Proc. 4th IEEE Conf. Autom. Sci. Eng.*, 2008, pp. 406–411.

[38] Y. Zhao, Q. Zhao, Q.-S. Jia, X. Guan, and X.-R. Cao, "Event-based optimization for dispatching policies in material handling systems of general assembly lines," in *Proc. 47th IEEE Conf. Decision Control*, 2008, pp. 2173–2178.

[39] A. Ferscha and S. K. Tripathi, "Parallel and distributed simulation of discrete event systems," in *Handbook of Parallel and Distributed Computing*. McGraw-Hill: New York, 1995.

[40] D. M. Nicol, "Parallel discrete-event simulation of FCFS stochastic queueing networks," in *Proc. ACM/SIGPLAN Conf. Parallel Programming: Experience with Applications, Languages and Systems*, 1988, pp. 124–137.

[41] C.-B. Yan, Q. Zhao, N. Huang, G. Xiao, and J. Li, "Line-side buffer assignment in general assembly line systems with material handling," in *Proc. 13th IFAC Symp. Inf. Control Prob. Manuf. (INCOM'09)*, 2009, pp. 1239–1244.

[42] V.-Y. Vee and W.-J. Hsu, "Parallel discrete event simulation: A survey", Centre for Advanced Information Systems, Nanyang Technical Univ., Nanyang, China, Report, 1998.

[43] D. M. Nicol, "Principles of conservative parallel simulation," in *Proc. 1996 Winter Simulation Conf.*, 1996, pp. 128–135.

[44] A. Park, R. M. Fujimoto, and K. S. Perumalla, "Conservative synchronization of large-scale network simulations," in *Proc. 18th Workshop on Parallel Distrib. Simulation*, 2004, pp. 153–161.

[45] A. Falcon, P. Faraboschi, and D. Ortega, "An adaptive synchronization technique for parallel simulation of networked clusters," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw.*, 2008, pp. 22–31.

[46] M. J. Chung and J. Xu, "And overhead reducing technique for time warp," *J. Parallel Distrib. Comput.*, vol. 65, pp. 65–73, 2005.

[47] D. W. Bauer, Jr. and E. H. Page, "Optimistic parallel discrete event simulation of the event-based transmission line matrix method," in *Proc. 2007 Winter Simulation Conf.*, 2007, pp. 676–684.

[48] R. Nelson and A. N. Tantawi, "Approximate analysis of fork/join synchronization in parallel queues," *IEEE Trans. Comput.*, vol. 37, pp. 739–743, 1988.

[49] H. W. Maalouf and M. K. Gurcan, "Minimisation of the update response time in a distributed database system," *Perform. Eval.*, vol. 50, pp. 245–266, 2002.

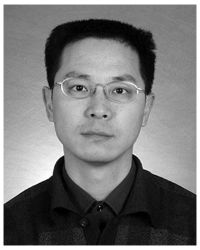
[50] Y. Zhao, C.-B. Yan, Q. Zhao, N. Huang, J. Li, and X. Guan, "Efficient simulation method for general assembly systems with material handling based on aggregated event-scheduling," Tech. Rep., 2009. [Online]. Available: http://www.cfins.au.tsinghua.edu.cn/files/paper/TASE309_full_20091003.pdf

Yanjia Zhao (S'06) received the B.S. degree in automatic control from Tsinghua University, Beijing, China, in 2005. He is currently working towards the Ph.D. degree at the Center for Intelligent and Networked Systems, Tsinghua University.

His research interests include modeling and optimizing complex systems, scheduling of manufacturing systems, and theory of Markov decision processes.

Chao-Bo Yan (S'06) received the B.S. degree from Xi'an Jiaotong University, Xi'an, China, in 2004. He is currently working towards the Ph.D. degree at the Center for Intelligent and Networked Systems (CFINS), Tsinghua University, Beijing, China.

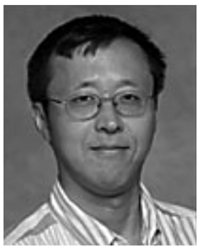
His research interests include simulation of the discrete-event dynamic systems (DEDS) and modeling, analysis, and optimization of production systems.



Qianchuan Zhao (M'06–SM'08) received the B.E. degree in automatic control, the B.S. degree in applied mathematics, and the Ph.D. degree in control theory and its applications from Tsinghua University, Beijing, China, in 1992, 1992, and 1996, respectively.

He is currently a Professor and Associate Director of the Center for Intelligent and Networked Systems (CFINS), Department of Automation, Tsinghua University. He was a Visiting Scholar at Carnegie Mellon University, Pittsburgh, PA, and Harvard University, Cambridge, MA, in 2000 and 2002, respectively. He was a Visiting Professor at Cornell University, Ithaca, NY, in 2006. His research interests include modeling, analysis, control and optimization for complex networked systems.

Prof. Zhao is an Associate Editor for the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERINGand the *Journal of Optimization Theory and Applications*.



Ningjian Huang received the Ph.D. degree in systems engineering from Oakland University, Rochester, MI, in 1991.

In 1991, he joined General Motors (GM) Corporation and has worked on numerous research projects in manufacturing related areas. Currently, he is Staff Engineer in the GM Research and Development Center. His research interests include manufacturing system modeling, simulation, analysis, and optimization.

Jingshan Li (S'97–M'00–SM'06) received the B.S. degree from the Department of Automation, Tsinghua University, Beijing, China, the M.S. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, and the Ph.D. degree in electrical engineering-systems from the University of Michigan, Ann Arbor, in 1989, 1992, and 2000, respectively.

From 2000 to 2006, he was a Staff Research Engineer at the Manufacturing Systems Research Laboratory, General Motors Research and Development Center, Warren, MI. He joined the University of Kentucky as an Assistant Professor in the Department of Electrical and Computer Engineering and the Center for Manufacturing in 2006. To date he has published about 80 refereed journal and conference papers. His primary research interests are in modeling, analysis and control of manufacturing, service, and health care systems.

Prof. Li received the 2009 IIE Transactions – Design and Manufacturing Best Application Paper Award, the 2005 IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING Best Paper Award, the 2006 IEEE Early Industry/Government Career Award in Robotics and Automation, and was also a finalists of the Best Paper Award of 2009 IEEE Conference on Automation Science and Engineering, and the Best Automation Paper Award of 2005 IEEE International Conference on Robotics and Automation. He was also the Associate Editor of the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, and *Mathematical Problems in Engineering*.

Xiaohong Guan (M'93–SM'95–F'07) received the B.S. and M.S. degrees in automatic control from Tsinghua University, Beijing, China, in 1982 and 1985, respectively, and the Ph.D. degree in electrical engineering from the University of Connecticut, Storrs, in 1993.

He was a Senior Consulting Engineer with PG&E from 1993 to 1995. From 1985 to 1988, and since 1995, he has been with the Systems Engineering Institute, Xian Jiaotong University, Xian, China, and currently, he is the Cheung Kong Professor of Systems Engineering, the Dean of School Electronic and Information Engineering, and Director of the Systems Engineering Institute. He is also the Director of the Center for Intelligent and Networked Systems and served as Head of the Department of Automation, Tsinghua University, from 2003 to 2008. He visited the Division of Engineering and Applied Science, Harvard University, from January 1999 to February 2000. His research interests include scheduling of power and manufacturing systems, bidding strategies for deregulated electric power markets, economy and security of complex network systems and sensor networks.