Formulation and a Simulation Based Algorithm for Line-side Buffer Assignment Problem in Systems of General Assembly Line with Material Handling

Chao-Bo Yan, *Student Member, IEEE,* Qianchuan Zhao, *Senior Member, IEEE,*
Ningjian Huang, Guoxian Xiao, and Jingshan Li, *Senior Member, IEEE*

***Abstract*—In systems of general assembly line with material handling, line-side buffers need to be carefully assigned to a limited number of material delivers (drivers) for part delivery to avoid production stoppage due to material shortage. Such a problem is referred to as Line-side Buffer Assignment Problem (LBAP). In this paper, we focus on fixed zoning version of LBAP. We formulate the problem, prove its NP-hardness, and propose an algorithm based on two structural characteristics of the LBAP problem–one being the analogousness between our problem and the Parallel Machine Scheduling (PMS) problem and the other being the monotonicity of the system throughput in the course of assigning line-side buffers to drivers. The developed algorithm globally converges with probability one when there exist feasible assignments. The algorithm is tested on a real system, and the results show that it is effective for solving the LBAP problem.**

***Note to Practitioners*—This paper was motivated by the needs to develop systematic methods to assign line-side buffers to a limited number of drivers in systems of general assembly line with material handling in automotive industry, such that the system throughput is maintained above a desired level. In practice, line-side buffers are typically assigned to drivers by the trial-and-error methods based on engineers' experience. In this paper, the fixed zoning version of the Line-side Buffer Assignment Problem (LBAP) is mathematically formulated. In order to solve the problem, two key structural characteristics of the problem are exploited and employed to design an algorithm. Numerical experiments on the real system show that the developed algorithm is effective for solving the LBAP problem. The flexible zoning version of the LBAP problem would be the future work.**

***Index Terms*—general assembly line, material handling, Line-side Buffer Assignment Problem (LBAP), NP-hard, Longest Processing Time (LPT) algorithm.**

This work was supported by a contract between Tsinghua University and General Motors Corporation. Qianchuan Zhao received additional support from NSFC (60574067, 60736027, 60721003), NCET program (NCET-04-0094), and the Program of Introducing Talents of Discipline to Universities (111 International Collaboration Project of China (B06002)), and Chao-Bo Yan received the additional support from China Scholarship Council (CSC). Partial of the work was reported at 13^{th} *IFAC Symposium on Information Control Problems in Manufacturing*, Moscow, 2009.

C.-B. Yan and Q. Zhao are with the Center for Intelligent and Networked Systems (CFINS), Department of Automation and TNList, Tsinghua University, Beijing 100084, China (e-mail: ycb04@mails.tsinghua.edu.cn and zhaoqc@tsinghua.edu.cn).

C.-B. Yan is also a visiting scholar at the Department of Electrical Engineering and Computer Science, the University of Michigan, Ann Arbor, MI 48109-2122, USA.

N. Huang and G. Xiao are with Research & Development Center, General Motors Corporation, Warren, MI 48090, USA (e-mail: ninja.huang, guoxian.xiao@gm.com).

J. Li is with Department of Electrical and Computer Engineering and Center for Manufacturing, University of Kentucky, Lexington, KY 40506, USA (e-mail: jingshan@engr.uky.edu).

Notations & Abbreviations

TP	Throughput of the system of general assembly line with material handling.
M	Number of machines in the assembly subsystem. All machines are indexed as m_u, $u = 1, 2, \ldots, M$, and all in-process buffers are indexed as b_u^I, $u = 0, 1, \ldots, M$.
τ_u	Cycle time of machine m_u, $u = 1, 2, \ldots, M$.
N_u^I	Number of in-process buffers upstream of machine m_u. All in-process buffers upstream of machine m_u are indexed as $b_{u_j}^I$, $j = 1, 2, \ldots, N_u^I$.
C_u^I	Capacity of in-process buffer b_u^I, $u = 1, 2, \ldots, M-1$.
N_u^L	Number of line-side buffers attached to machine m_u, $u = 1, 2, \ldots, M$. All line-side buffers attached to machine m_u are indexed as b_{uv}^L, $v = 1, 2, \ldots, N_u^L$, and $\mathbb{B} = \bigcup_{u=1}^{M} \bigcup_{v=1}^{N_u^L} \{b_{uv}^L\}$ is the set of all line-side buffers in the system.
C_{uv}^L	Capacity of line-side buffer $b_{uv}^L \in \mathbb{B}$.
USG_{uv}	Average usage rate (consumption number) of parts in line-side buffer $b_{uv}^L \in \mathbb{B}$ consumed by the job in a cycle.
QTY_{uv}	Average delivery quantity of parts for line-side buffer $b_{uv}^L \in \mathbb{B}$ when it is served in the trip.
RTT_{uv}	Average round trip time to finish part delivery from the central warehouse to line-side buffer $b_{uv}^L \in \mathbb{B}$ and return to the central warehouse.
D	Number of drivers in the material handling subsystem. All drivers are indexed as d_i, $i = 1, 2, \ldots, D$.
π	Delivery policy for drivers' part delivery.
A	(Partial) assignment of line-side buffers to drivers, $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, where $\mathbb{A}_i$ is the set of line-side buffers assigned to driver d_i, $i = 1, 2, \ldots, D$.
LBAP	Line-side Buffer Assignment Problem.
PMS	Parallel Machine Scheduling.
LPT	Longest Processing Time.

I. Introduction

In production line design, material handling is a very important element in terms of increasing system efficiency and reducing manufacturing costs. In some production systems such as in automotive industry, material handling refers to drivers' delivering parts to the general assembly line such that operators would use the parts for assembly without walking long distances. The parts being delivered will be stored in line-side buffers with limited capacities. The long time shortage

of parts in one or more line-side buffers (because of no drivers delivering parts for them in time) will cause production stoppage and result in low system throughput. Thus, material handling has great impact on production, as well as on cost. In fact, it is estimated that material handling accounts for 20–50% cost in manufacturing systems [1]. Since line-side buffers are assigned to drivers for part delivery, the purpose is to find an assignment of the line-side buffers to the given number of drivers such that the system throughput is maintained above a desired level. The corresponding problem will be called the *Line-side Buffer Assignment Problem* (LBAP). In practice, line-side buffers are typically assigned to drivers by trial-and-error methods based on engineers' experience. Although the manual assignment works fine when the number of drivers is relatively large, systematic methods are needed when the number of drivers is close to the lean number (i.e., the minimum number of drivers allowing an assignment under which the throughput is above the desired level) as the LBAP problem becomes very hard in these cases. In this paper, we focus on line-side buffer assignment methods based on the system model and structural characteristics. Clearly, a driver's zoning (i.e., the set of line-side buffers in the charge of the driver) could be either fixed or flexible in the system and flexible zoning strategy is superior to the fixed one in the sense of timely part delivery. However, fixed zoning strategy has its advantages of reducing errors in the delivery process and saving cost for drivers' cross training. Thus, in most cases, drivers' zonings are preferred to be fixed unless the flexible zoning strategy is indispensable. In this paper, we concentrate on the fixed zoning strategy.

A schematic view of the system of general assembly line with material handling is given in Fig. 1. In Fig. 1, the circles represent machines on which jobs are processed (or assembled). The rectangles marked with b_u^I represent in-process buffers where work-in-process jobs are held; those marked with b_{uv}^L represent line-side buffers, each storing one type of parts for assembly. The trapezoid in Fig. 1 represents a central warehouse storing all types of parts needed by the line-side buffers; the ellipses represent drivers who deliver parts from the central warehouse to the line-side buffers. In the system of general assembly line with material handling, the machines and the in-process buffers form the *assembly subsystem*, and the central warehouse and the drivers form the *material handling subsystem*; these two subsystems are connected by the line-side buffers. Details about these two subsystems can be found in Subsection II-A.

There is a lot of research on production lines in literature. Various production lines in manufacturing systems are investigated in [2]. [3], [4], and [5] established simulation models to evaluate the performance of complex production lines. In addition to simulation models, some analytical approximation methods are also developed for this purpose [6] [7] [8]. For design and optimization of production lines, in-process buffer space allocation problem [9] [10] and assembly line balancing problem [11] [12] are extensively studied.

Although the LBAP problem is important and a lot of research contributes to analysis, design, and optimization of production lines, there seems surprisingly little literature investigating the LBAP problem and no method but the trial-and-error methods in practice is reported to address this issue. The challenges in solving the LBAP problem come from two aspects. Firstly, the problem has its root as a complex combinatorial problem as we will see in Section III. Even when the machines are reliable, we still have to face the huge solution space with few structural properties that allow us to solve the problem in polynomial time. Secondly, the complex system dynamics and the randomness in the production process make the problem even harder since the system throughput has no precise and analytical expressions and thus the feasibility checking is time-consuming. It should be pointed out that analytical approximation methods such as decomposition method and aggregation method developed in [6] and [7] for evaluating the throughput and other system performance measures need substantial extensions before could be applied for the setting in this paper. The reason is the correlation among the operations of different machines introduced by shared drivers.

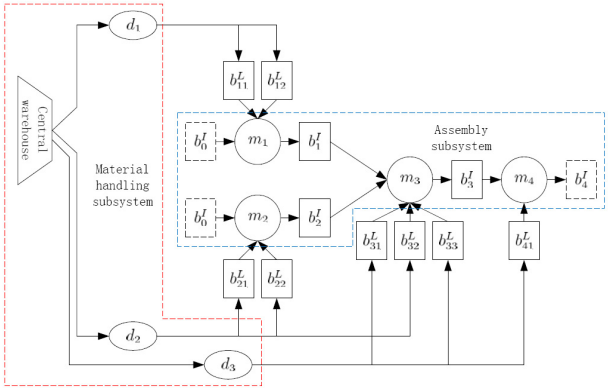


Fig. 1. A schematic view of the system of general assembly line with material handling

To our best knowledge, there is no available effective method to solve the LBAP problem. Therefore, the goal of this paper is to contribute to this end. In the following, we will briefly review three classes of problems in literature related to our study. The first class of problems is the Assignment Problem (AP). AP and its generalized models are well defined and studied in literature, such as [13] and [14]. In the AP problem, the cost of assigning a task to a worker is additive. Nevertheless, the system throughput, which depends on the whole assignment of line-side buffers to drivers in the LBAP problem, is not additive. We cannot apply AP algorithms to solve the LBAP problem because the additivity is necessary in AP algorithms. The second class is the NP-complete problem–the Bin-Packing Problem (BPP) where the number of bins is minimized [15]. Although the LBAP problem seems similar to the BPP problem, BPP algorithms are not suitable to solve the LBAP problem because of the difficulty of deciding "bin capacity" due to the complex impact of the system dynamics on the system throughput. However, the third class of problems–the Parallel Machine Scheduling (PMS) problem [16], which is in a sense regraded as the dual problem to the BPP problem [17], can give us some insights into the methods to solve our problem, which will be demonstrated in Sections

IV and V. The PMS problem, where the makespan (i.e., the latest completion time of all tasks) is minimized, is also NP-complete [18] (in [18], PMS is called multiprocessor scheduling problem). We will use the PMS problem to establish our complexity result for the LBAP problem, which is not related to the PMS problem at the first glance. Due to analogousness of these two problems (will be developed in Subsection V-A), we will use PMS algorithms to reduce the search space in the first phase of our algorithm. Although the PMS problem can be viewed as a special case of the LBAP problem after a suitable transformation, it should be pointed out that our problem is harder than it since we have the additional difficulty caused by the complex system dynamics and the existence of randomness in the production process which make the feasibility checking of our problem very time-consuming. As usual, we will appeal to simulation models to make the feasibility checking. Since the problem is computationally intractable and the evaluation of the throughput provided by the simulation model is noisy, inspired by the Nested Partitions Algorithm [19] [20], we will introduce backtracking in the second phase of our algorithm.

The main contributions of this paper are to formulate the fixed zoning version of the LBAP problem, prove its NP-hardness by polynomial-time reduction from the Parallel Machine Scheduling (PMS) problem, and develop an algorithm, which can effectively assign line-side buffers to drivers. The global convergence of the algorithm is established when feasible assignments exist. It is worth noting that, although the focus of this paper is on the general assembly line, the formulation, the complexity result, and the solution methodology can be easily extended to systems containing disassembly and/or rework operations.

The remainder of the paper is organized as follows. In Section II, we will define the system model of the general assembly line with material handling and formulate the LBAP problem under the fixed zoning assumption. In Section III, we will prove the NP-hardness of the problem. In Section IV, we will propose an algorithm for the LBAP problem based on its two structural characteristics. In Section V, the two structural characteristics employed in the algorithm will be investigated and the global convergence of the algorithm will be established. In Section VI, we will examine the rationality and the effectiveness of the algorithm by numerical experiments. Finally, Section VII will conclude the context. Notations for the system state and proofs for the propositions in Subsections V-B and V-C are respectively presented in Appendices A and B.

II. Problem Formulation

In this section, the Line-side Buffer Assignment Problem (LBAP) will be mathematically formulated under the fixed zoning assumption. First, details about the assembly subsystem and the material handling subsystem will be described in Subsection II-A. Then, the model of the system will be defined in Subsection II-B and the LBAP problem will be formulated in Subsection II-C. Finally, the dynamics of the system will be developed in Subsection II-D.

A. System Description

As we mentioned in the Introduction, the system of general assembly line with material handling consists of the assembly subsystem and the material handling subsystem, which are connected by the line-side buffers. In the following, these two subsystems will be separately described in detail.

As shown in Fig. 1, the assembly subsystem is a tree-structured system, in which machines and in-process buffers are alternately connected. The assembly subsystem has at least one input in-process buffer and only one output in-process buffer (they are marked as dashed rectangles in Fig. 1). Except for the input and the output in-process buffers, capacities of all other in-process buffers are *finite*. The normal operation time for the job on a machine is referred to as *cycle time* of the machine. All machines are *unreliable*. This feature of the machine is characterized by two random variables: the time between failures (also called failure time) and the time to repair (also called repair time). Expectations of them are Mean Time Between Failures (MTBF) and Mean Time To Repair (MTTR) respectively. When a machine operates, one job is taken from each upstream in-process buffer and simultaneously a given number of parts from each line-side buffer attached to the machine; as soon as the operation is completed, one job is put into the downstream in-process buffer. If a machine is *operational* (namely, it is up and busy), it will be *starved for jobs* if at least one upstream in-process buffer is empty and be *blocked by jobs* if the downstream in-process buffer is full, and be *starved for parts* if at least one of the line-side buffers attached to it is short of parts; any one of these three kinds of events will make the machine *idle* (namely, it is up but forced down). When a machine breaks down or is idle, its production capacity is lost.

In the system, the material handling subsystem is responsible for part replenishment for the line-side buffers. In the material handling subsystem, drivers follow a *delivery policy* to deliver parts for the line-side buffers. The delivery policy specifies for each driver which line-side buffer in the charge of him/her to serve in the next trip and when to depart from the central warehouse for the service. As soon as the parts are delivered into the line-side buffer, the driver returns to the central warehouse and will either start his/her next trip for part delivery right away or be idle at the warehouse, depending on the action determined by the delivery policy.

B. System Model

We make the following assumptions on the system of general assembly line with material handling.

1) The assembly subsystem is a tree-structured system. The input in-process buffers have inexhaustible supplies for the downstream machines and the output in-process buffer has an inexhaustible stock for the finished products coming from its upstream machine. Excluding the input and the output in-process buffers, in the system, there are M machines and $M-1$ in-process buffers alternately connected with each other.
2) The cycle time of machine m_u is deterministic and denoted by τ_u, $u = 1, 2, \ldots, M$. All machines are

unreliable. The repair starts as soon as a machine breaks down. If repaired, the machine immediately resumes the operation of processing the interrupted job. Jobs are not scrapped or rejected on any machine.

3) The capacity of in-process buffer b_u^I is finite and denoted as C_u^I, $0 < C_u^I < \infty$, $u = 1, 2, \ldots, M-1$.
4) For line-side buffer $b_{uv}^L \in \mathbb{B}$, on the average, USG_{uv} parts are consumed by the job in a cycle and if the line-side buffer is served by the driver, QTY_{uv} parts are delivered in a trip, and the driver's average round trip time from the central warehouse to the line-side buffer is RTT_{uv}, where $\mathbb{B} = \bigcup_{u=1}^{M} \bigcup_{v=1}^{N_u^L} \{b_{uv}^L\}$ is the set of all line-side buffers in the system. The capacity of line-side buffer b_{uv}^L is C_{uv}^L, which is huge enough so that when b_{uv}^L is empty, any batch of parts being delivered in a trip can be put into it. Inexhaustible supplies of all types of parts are available in the central warehouse.
5) There are D drivers in the material handling subsystem. Their capabilities are identical and each line-side buffer is eligible to be assigned to every one of them. All line-side buffers must be assigned to the drivers and drivers' zonings are fixed and non-overlapping. The assignment of line-side buffers to drivers is defined in Definition 2.3.
6) All drivers are idle at the central warehouse at the beginning of the production. They follow a delivery policy to deliver parts and each can deliver parts for only one line-side buffer in a trip. The delivery policy is defined in Definition 2.4. In the delivery process, if the spare space in the line-side buffer is not enough for the parts being delivered, the driver will wait until all parts can be put into it.

Remark 2.1: Note that 1)–3) define the assembly subsystem or the system of general assembly line without material handling. These assumptions are prevalently made in transfer line or assembly/disassembly line literature, such as in [6] and [7]. Assumptions 4)–6) define the material handling subsystem. Clearly, if all line-side buffers have sufficient part supply, the system of general assembly line with material handling will degrade into a system without material handling.

Remark 2.2: The assumption of tree-structured assembly subsystem in 1) and that the capacity of the line-side buffer is huge enough in 4) are just to simplify the system dynamics. Removing them will not affect the formulation, the complexity result, and the solution for the LBAP problem.

Remark 2.3: Clearly, in the long run, the average numbers of jobs produced on all machines are the same. The model can be easily extended to systems in which the average numbers of jobs produced on machines are in proportion. For example, if the fraction of scrapped jobs on a machine is 1% and independent of the production process, the formulation and the solution methodology for the LBAP problem remain the same if average usage rates of parts in line-side buffers before or attached to this machine are increased by 1.01%, which implies 2) can be more general. Clearly, this method also works in the case that defective parts considered in the system.

C. Problem Definition

Before formulating the Line-side Buffer Assignment Problem (LBAP), we first define the throughput, the assignment, and the delivery policy, which will be involved in the formulation of the problem.

Definition 2.1: Throughput TP, measured in jobs per hour (jph), is the average number of finished products arriving at the output in-process buffer in unit time in the steady state of the system. Mathematically,

$$\mathrm{TP} = \lim_{T \to \infty} \frac{E_\xi[K_T(\xi)]}{T}, \tag{1}$$

where $K_T(\xi)$ is the number of finished products arriving at the output in-process buffer in a period of time T and ξ is all randomness in the system which consists of random failure and random repair of machines, random part consumption and random quantity of part delivery in line-side buffers, and random round trip time for drivers' delivering parts. Details about the randomness can be found in Subsection II-D.

Definition 2.2: Driver's utilization Γ is his average service time for part delivery in unit time in the steady state of the system. Mathematically,

$$\Gamma_i = \lim_{T \to \infty} \frac{E_\xi[T_i(\xi)]}{T}, \ i = 1, 2, \ldots, D, \tag{2}$$

where $T_i(\xi)$ is driver d_i's total service time in a period of time T, ξ is all randomness in the system, and D is the number of drivers in the material handling subsystem.

Remark 2.4: Obviously, based on the definition, all drivers' utilizations are not beyond 1.

Definition 2.3: An assignment A is a fixed partition $(\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$ of set $\mathbb{B}$ of all line-side buffers into D disjoint sets, one for each driver. Mathematically, for assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, $\mathbb{A}_i \cap \mathbb{A}_{i'} = \emptyset$, $\forall i, i' = 1, 2, \ldots, D$, $i \neq i'$, and $\bigcup_{i=1}^{D} \mathbb{A}_i = \mathbb{B}$.

For example, in the system shown in Fig. 1, the assignment of line-side buffers to drivers is $A = (\mathbb{A}_1, \mathbb{A}_2, \mathbb{A}_3, \mathbb{A}_4)$, where $\mathbb{A}_1 = \{b_{33}^L, b_{51}^L, b_{52}^L, b_{53}^L\}$, $\mathbb{A}_2 = \{b_{31}^L, b_{32}^L\}$, $\mathbb{A}_3 = \{b_{11}^L, b_{12}^L, b_{21}^L\}$, $\mathbb{A}_4 = \{b_{22}^L, b_{61}^L\}$.

To define the delivery policy, the state space of the system and the driver's action space are defined in the following.

X_t State vector of the system at time instant t, $t \geqslant 0$, i.e., $X_t = (s_t^m, s_t^{Ib}, s_t^{Lb}, s_t^d)$, where s_t^m is the state of the machines, s_t^{Ib} is the state of the in-process buffers, s_t^{Lb} is the state of the line-side buffers, and s_t^d is the state of the drivers. These notations are defined in Appendix A in detail.

a_t Driver's action vector at time instant t, $t \geqslant 0$. It is a D-by-1 vector whose element $a_{i,t}$ is driver d_i's action at t; specifically, if driver d_i is idle and to serve line-side buffer $b_{uv}^L \in \mathbb{A}_i$ at t, then $a_{i,t} = b_{uv}^L$, otherwise $a_{i,t} = a^*$, $i = 1, 2, \ldots, D$, where a^* is a virtual action which has no effect on the corresponding driver.

$\mathcal{S}$ State space of the system. $\mathcal{S} := \{\text{all } X_t\}$.

$\mathcal{A}$ Driver's action space. $\mathcal{A} := \{\text{all } a_t\}$.

Definition 2.4: A decision rule π_t, which specifies all drivers' actions at time instant t, is a mapping from the state space of the system to the driver's action space, i.e.,

$\pi_t : \mathcal{S} \mapsto \mathcal{A}$ and $a_t = \pi_t(X_t)$. A delivery policy π is a sequence of decision rules, i.e., $\pi = (\pi_t)_{t \geqslant 0}$.

Remark 2.5: Let $T^{dp}_{i,b^L_{uv}}(l)$ and $T^{rt}_{i,b^L_{uv}}(l)$ respectively denote driver d_i's departure time from and return time to the central warehouse for part delivery for line-side buffer b^L_{uv} in his/her l^{th} trip. If decision rule π_t specifies serving line-side buffer b^L_{uv} in driver d_i's l^{th} trip, then $a_{i,t} = (\pi_t(X_t))_i = b^L_{uv}$ and $T^{dp}_{i,(\pi_t(X_t))_i}(l) = T^{dp}_{i,b^L_{uv}}(l) = t$, $t \geqslant 0$. From Definition 2.4, for all delivery policies, we have $T^{rt}_{i,b^L_{u'v'}}(l-1) \leqslant T^{dp}_{i,b^L_{uv}}(l)$. In this remark, $b^L_{u'v'}, b^L_{uv} \in \mathbb{A}_i$, $i = 1, 2, \ldots, D$, $l = 1, 2, \ldots$

Remark 2.6: The system throughput and drivers' utilizations depend on the assignment and is also affected by the delivery policy. Thus, unless otherwise noted we will use $\mathrm{TP}^\pi(A)$ and $\Gamma^\pi(A)$ to denote the system throughput and the driver's utilization respectively hereafter. Under a delivery policy π, the assignment A is feasible if and only if $\mathrm{TP}^\pi(A) \geqslant \mathrm{TP}_0$, where TP_0 is the desired system throughput; otherwise the assignment is infeasible. It should be pointed out that due to the complexity in the system dynamics (will be developed in Subsection II-D), the analytical expression of $\mathrm{TP}^\pi(A)$ is extremely hard to obtain if not totally impossible. This makes the feasibility checking of an assignment very difficult and only numerical methods can be used.

As an example, let us look at a simple delivery policy known as the Reorder Point (RP) delivery policy π^{RP}, which is usually used in the material handling subsystem because it is easy to implement on the factory floor. The RP delivery policy π^{RP} consists of two processes: reorder and delivery. The reorder process is as follows: when the inventory level of a line-side buffer $b^L_{uv} \in \mathbb{B}$ is equal to or lower than a reorder level RL_{uv} $(= \left\lceil \frac{\mathrm{USG}_{uv} \cdot \mathrm{RT}}{\tau_u} \right\rceil)$, b^L_{uv} will send a request to corresponding driver for part delivery as soon as the operation on machine m_u is completed, where RT is called the reorder threshold and is a fixed length of time, say 15 minutes. Line-side buffer b^L_{uv} can't reorder again even if its inventory level is still lower than the reorder level RL_{uv} unless the requested parts are delivered. The delivery process is as follows: the driver responds requests according to the First Come First Served (FCFS) discipline; if there is at least one request waiting for response, the driver will respond the first one as soon as he returns to the central warehouse; if more than one request are sent to a driver at the same time, the tie can be broken arbitrarily. The reorder and the delivery processes are shown in Fig. 2, where line-side buffer b^L_{uv} is in the charge of driver d_i and $T^{ro}_{i,b^L_{uv}}$ is its reorder time. The way in which the RP delivery policy specifies drivers' actions is presented in Example 2.1 in Subsection II-D3.

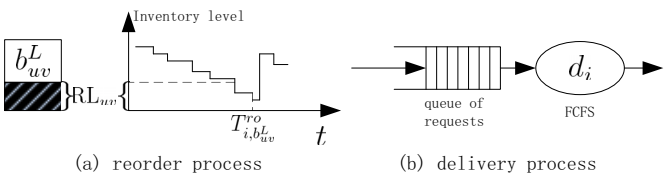


Fig. 2. The Reorder Point (RP) delivery policy

Based on these definitions, the problem to assign line-side buffers to drivers such that the throughput is maintained above a desired level TP_0 can be formulated as follows.

Problem LBAP: In the system of general assembly line with material handling, find an assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$ of line-side buffers to D drivers,

$$\text{s.t.:} \quad \mathrm{TP}^\pi(A) \geqslant \mathrm{TP}_0, \tag{3}$$

$$\bigcup_{i=1}^{D} \mathbb{A}_i = \mathbb{B}, \tag{4}$$

$$\mathbb{A}_i \cap \mathbb{A}_{i'} = \emptyset, \ \forall i, i' = 1, 2, \ldots, D, \ i \neq i', \tag{5}$$

where π is a given delivery policy and $\mathbb{B}$ is the set of all line-side buffers.

Clearly, in the system, if line-side buffers b_{uv} and $b_{u'v'}$ cannot be assigned to the same driver, we can add the following constraint in the formulation:

$$b_{uv} \in \mathbb{A}_i, b_{u'v'} \notin \mathbb{A}_i, \forall i = 1, 2, ..., D. \tag{6}$$

The LBAP problem formulated above is a decision problem, or called constraint satisfaction problem, as described in [18]. There is no objective function in a decision problem. The problem is solved either when a solution meeting all constraints is found or it is proved that there is no feasible solution.

D. System Dynamics

Under a given delivery policy π and an assignment A of line-side buffers to drivers, the throughput of the system can be determined. To do so, we have to specify system dynamics. In this study, the system dynamics are developed by a timing-focused method, in which only key timings (timings that can reproduce the system dynamics and figure out other timings in the system) need to be investigated in the system dynamics. The dynamics of the assembly subsystem and the material handling subsystem are separately developed in Subsections II-D1 and II-D2, and that of their interactions through the line-side buffers is developed in Subsection II-D3.

1) Assembly Subsystem: For any machine in the assembly subsystem, there are one or more in-process buffers upstream of it and an in-process buffer downstream of it. Thus, the assembly subsystem can be viewed as a series of machine-centered segments. The u^{th} segment, in which there are N^I_u in-process buffers upstream of machine m_u and N^L_u line-side buffers attached to machine m_u, $u = 1, 2, \ldots, M$, is shown in Fig. 3.

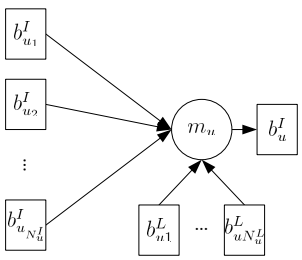


Fig. 3. A general configuration of a segment of the assembly subsystem

For the u^{th} segment of the assembly subsystem, $u = 1, 2, \ldots, M$, the key timings of events are:

1) exit time $T^e_{u_j}(k)$ of the k^{th} job from the upstream in-process buffer $b^I_{u_j}$, $j = 1, 2, \ldots, N^I_u$, $k = 1, 2 \ldots$
2) release time $T^r_{uv}(k)$ of all parts needed by the k^{th} job in line-side $b^L_{uv} \in \mathbb{B}$, $k = 1, 2 \ldots$
3) start time $T^s_u(k)$ of processing/assembling the k^{th} job on machine m_u, $k = 1, 2 \ldots$
4) finish time $T^f_u(k)$ of processing/assembling the k^{th} job on machine m_u, $k = 1, 2 \ldots$
5) arrival time $T^a_u(k)$ of the k^{th} job at the downstream in-process buffer b^I_u (the departure time from machine m_u), $k = 1, 2, \ldots$

Other related timing is:

1) release time $T^{pr}_{uv}(q)$ of the q^{th} part in line-side buffer $b^L_{uv} \in \mathbb{B}$, i.e., the arrival time of the part at the line-side buffer, $q = 1, 2, \ldots$

Now, we can characterize the dynamics of the assembly subsystem by a group of recursive equations as follows:

$$\begin{aligned} T^e_{u_j}(k) &= \max(T^a_{u_j}(k), T^a_u(k-1)), \\ &\quad j = 1, 2, \ldots, N^I_u,\ u = 1, 2, \ldots, M, \\ &\quad k = 1, 2, \ldots, \end{aligned} \tag{7}$$

$$\begin{aligned} T^r_{uv}(k) = &\max_{\sum_{j=1}^{k-1} Q^C_{j,uv} < q \leqslant \sum_{j=1}^{k} Q^C_{j,uv}} T^{pr}_{uv}(q), \\ &b^L_{uv} \in \mathbb{B},\ k = 1, 2, \ldots, \end{aligned} \tag{8}$$

$$\begin{aligned} T^s_u(k) = \max(&\max_{1 \leqslant j \leqslant N^I_u} T^e_{u_j}(k), \max_{1 \leqslant v \leqslant N^L_u} T^r_{uv}(k)), \\ &u = 1, 2, \ldots, M,\ k = 1, 2, \ldots, \end{aligned} \tag{9}$$

$$\begin{aligned} T^f_u(k) &= T^s_u(k) + T^p_u(k), \\ &u = 1, 2, \ldots, M,\ k = 1, 2, \ldots, \end{aligned} \tag{10}$$

$$\begin{aligned} T^a_u(k) &= \max(T^f_u(k), T^e_u(k - C^I_u)), \\ &u = 1, 2, \ldots, M,\ k = 1, 2, \ldots \end{aligned} \tag{11}$$

The randomness of the system is in part included in (8) and (10). Specifically, in (8), $Q^C_{j,uv}$, the quantity of parts needed by the j^{th} job in line-side buffer $b^L_{uv} \in \mathbb{B}$, is equal to an integer random variable $\xi_{3,uv}$ with mean USG_{uv}, $j = 1, 2, \ldots$ In (10), $T^p_u(k)$ is the processing/assembling time of the k^{th} job on machine m_u including repair times of the machine (if any), $u = 1, 2, \ldots, M$, $k = 1, 2, \ldots$ In other words, if machine m_u does not break down during processing/assembling the job k, the processing time is the cycle time τ_u; otherwise total repair time, namely, total downtime, during the processing period, must be included. Mathematically,

$$T^p_u(k) = \tau_u + t((\xi_{1,u}(l))_{l \geqslant 1}, (\xi_{2,u}(l))_{l \geqslant 1}), \tag{12}$$

where $t((\xi_{1,u}(l))_{l \geqslant 1}, (\xi_{2,u}(l))_{l \geqslant 1})$ is total repair time of machine m_u during processing k^{th} job, $\xi_{1,u}(l)$ and $\xi_{2,u}(l)$ are its l^{th} failure time and repair time, and $\xi_{1,u}(l)$'s (correspondingly $\xi_{2,u}(l)$'s) are identically and independently distributed. Readers interested in the generation of processing times are referred to [21].

In (7)–(11), blocking mechanism of Blocking-After-Service (BAS) as in [6] is assumed. As for Blocking-Before-Service (BBS) as in [7], the first and the last equations are as follows:

$$\begin{aligned} T^e_{u_j}(k) &= \max(T^a_{u_j}(k), T^a_u(k-1), T^e_u(k - C^I_u)), \\ &\quad j = 1, 2, \ldots, N^I_u,\ u = 1, 2, \ldots, M, \\ &\quad k = 1, 2, \ldots, \end{aligned} \tag{13}$$

$$T^a_u(k) = T^f_u(k),\ u = 1, 2, \ldots, M,\ k = 1, 2, \ldots \tag{14}$$

In the assembly subsystem, we assume at the beginning of the production, there is no job on any machine or in any in-process buffer except for the input in-process buffers and no machine breaks down. Therefore, the initial conditions of the assembly subsystem are:

$$T^a_0(k) = 0,\ k = 1, 2, \ldots, \tag{15}$$

$$T^a_u(0) = 0,\ u = 1, 2, \ldots, M, \tag{16}$$

$$T^e_u(k) = 0,\ k \leqslant 0,\ u = 1, 2, \ldots, M. \tag{17}$$

The first condition implies arrival times of all jobs at input in-process buffer b^I_0's are 0, the latter two indicate there is no job on any machine or in any other in-process buffer. Based on (7)–(14) and the initial conditions, the dynamics of the assembly subsystem can be reproduced.

2) Material Handling Subsystem: In the material handling subsystem, parts are delivered by the drivers. Thus, the process of driver's delivering parts for the line-side buffers can be reproduced if timings of key events related to the drivers are determined. The parts being delivered for line-side buffer $b^L_{uv} \in \mathbb{A}_i$ in driver d_i's l^{th} trip are indexed as $Q^D_{l-1,uv} + 1, Q^D_{l-1,uv} + 2, \ldots, Q^D_{l,uv}$, where $Q^D_{l-1,uv}$ is the quantity of parts delivered in line-side buffer b^L_{uv} before driver d_i's l^{th} trip and $Q^D_{l,uv} - Q^D_{l-1,uv}$ is equal to an integer random variable $\xi_{4,uv}$ with mean QTY_{uv}, $i = 1, 2, \ldots, D$, $l = 1, 2, \ldots$, and D is the number of drivers in the material handling subsystem. Then for the i^{th} driver d_i in this subsystem, $i = 1, 2, \ldots, D$, the key timings of events are:

1) driver d_i's departure time $T^{dp}_{i,b^L_{uv}}(l)$ from the central warehouse to deliver parts for line-side buffer $b^L_{uv} \in \mathbb{A}_i$ in his/her l^{th} trip, $l = 1, 2, \ldots$
2) driver d_i's part delivery time $T^{pd}_{i,b^L_{uv}}(l)$ at line-side buffer $b^L_{uv} \in \mathbb{A}_i$ in his/her l^{th} trip (i.e., the release times of parts being delivered), $l = 1, 2, \ldots$
3) driver d_i's return time $T^{rt}_{i,b^L_{uv}}(l)$ to the central warehouse from the served line-side buffer $b^L_{uv} \in \mathbb{A}_i$ in his/her l^{th} trip, $l = 1, 2, \ldots$

Other related timing is:

1) part consumption time $T^{pc}_{uv}(q)$ of the q^{th} part in line-side buffer $b^L_{uv} \in \mathbb{B}$, $q = 1, 2, \ldots$

Thus, under a delivery policy π, part delivery for line-side buffer $b^L_{uv} \in \mathbb{A}_i$ in driver d_i's l^{th} trip can be characterized as follows:

$$\begin{aligned} T^{pd}_{i,b^L_{uv}}(l) = \max(&T^{dp}_{i,b^L_{uv}}(l) + \frac{T^T_{l,uv}}{2}, \\ &T^{pc}_{uv}(Q^D_{l,uv} - C^L_{uv})), \\ &b^L_{uv} \in \mathbb{A}_i,\ i = 1, 2, \ldots, D,\ l = 1, 2, \ldots \end{aligned} \tag{18}$$

$$T^{rt}_{i,b^L_{uv}}(l) = T^{pd}_{i,b^L_{uv}}(l) + \frac{T^T_{l,uv}}{2}, \quad b^L_{uv} \in \mathbb{A}_i,\ i = 1,2,\ldots,D,\ l = 1,2,\ldots \tag{19}$$

where $T^T_{l,uv}$, driver d_i's round trip time for serving line-side buffer b^L_{uv} in his/her l^{th} trip, is equal to a random variable $\xi_{5,uv}$ with mean RTT_{uv}.

In addition to these equations, following initial condition is needed to characterize the dynamics of the material handling subsystem:

$$T^{pc}_{uv}(q) = 0,\ b^L_{uv} \in \mathbb{B},\ q \leqslant 0. \tag{20}$$

This condition implies that all line-side buffers are empty at the beginning of the production. In the material handling subsystem, the maximum delivery quantity of parts for line-side buffer $b^L_{uv} \in \mathbb{B}$ in a trip is not more than the capacity C^L_{uv}. Thus, from (18) and considering (20), we know the spare space is enough for parts delivered in the first delivery for every line-side buffer. Therefore, together with (20), the recursive equations can reproduce the part delivery process in the material handling subsystem.

3) Interaction Between the Two Subsystems: The assembly subsystem and the material handling subsystem are connected by the line-side buffers. In the assembly subsystem, the machines take parts from the line-side buffers; in the material handling subsystem, the drivers deliver parts into the line-side buffers. These two subsystems cooperate smoothly so that the production goes on and on. They interact with each other as follows.

For parts indexed as $Q^D_{l-1,uv}+1, Q^D_{l-1,uv}+2, \ldots, Q^D_{l,uv}$ and delivered in driver d_i's l^{th} trip for line-side buffer b^L_{uv}, we have

$$T^{pr}_{uv}(q) = T^{pd}_{i,b^L_{uv}}(l), \quad q = Q^D_{l-1,uv}+1, Q^D_{l-1,uv}+2, \ldots, Q^D_{l,uv}, \quad b^L_{uv} \in \mathbb{A}_i,\ i = 1,2,\ldots,D,\ l = 1,2,\ldots \tag{21}$$

And for part consumption, we have

$$T^{pc}_{uv}(q) = T^s_u(k), \quad q = \sum_{j=1}^{k-1} Q^C_{j,uv}+1, \sum_{j=1}^{k-1} Q^C_{j,uv}+2, \ldots, \sum_{j=1}^{k} Q^C_{j,uv},\ b^L_{uv} \in \mathbb{B},\ k = 1,2,\ldots \tag{22}$$

Besides these two equations, the delivery policy π may also make these two subsystems interact with each other. The Reorder Point delivery policy π^{RP} is exemplified to develop this aspect of the system dynamics in the following.

Example 2.1 (The RP delivery policy π^{RP}): Let $T^{ro}_{i,b^L_{uv}}(l)$ denote the reorder time of the line-side buffer $b^L_{uv} \in \mathbb{A}_i$ served in driver d_i's l^{th} trip, $i = 1,2,\ldots,D$, $l = 1,2,\ldots$ Thus, under the RP delivery policy π^{RP}, if the number of parts in line-side buffer b^L_{uv} is more than RL_{uv} after the $(k-1)^{th}$ job being processed and is not after the k^{th} job, then:

$$T^{ro}_{i,b^L_{uv}}(l) = T^f_u(k), \quad b^L_{uv} \in \mathbb{A}_i,\ i = 1,2,\ldots,D, \quad k = 1,2,\ldots,\ l = 1,2,\ldots \tag{23}$$

Recall that the reorder process is shown in Fig. 2. The driver's departure time from the central warehouse to the line-side buffer for part delivery is as follows:

$$T^{dp}_{i,b^L_{uv}}(l) = \max(T^{rt}_{i,b^L_{u'v'}}(l-1), T^{ro}_{i,b^L_{uv}}(l)), \quad b^L_{u'v'}, b^L_{uv} \in \mathbb{A}_i, \quad i = 1,2,\ldots,D,\ l = 1,2,\ldots \tag{24}$$

At the beginning of the production, all drivers are assumed to be available at the central warehouse, i.e., $T^{rt}_{i,b^L_{u'v'}}(0) = 0$.

Obviously, π^{RP} is a delivery policy because $T^{rt}_{i,b^L_{u'v'}}(l-1) \leqslant T^{dp}_{i,b^L_{uv}}(l)$ and it specifies driver d_i's action: at time $t = T^{dp}_{i,b^L_{uv}}(l)$, $a_{i,t} = (\pi^{\mathrm{RP}}_t(X_t))_i = b^L_{uv}$; otherwise $a_{i,t} = a^*$, where a^* is a virtual action which has no effect on the corresponding driver, $b^L_{u'v'}, b^L_{uv} \in \mathbb{A}_i$, $i = 1,2,\ldots,D$, $l = 1,2,\ldots$

III. NP-HARDNESS OF LBAP

In this section, we will address the complexity result for the Line-side Buffer Assignment Problem (LBAP) defined in Section II. This result is established by polynomial-time reduction from the NP-complete problem–the Parallel Machine Scheduling (PMS) problem. For our purposes, we first describe the PMS problem.

The PMS problem is stated as follows: does there exist a schedule of n tasks with processing time $l_v > 0$ for the v^{th} task, $v = 1,2,\ldots,n$, onto k parallel processors, such that the makespan (latest completion time of all tasks) is no more than a given time c_0? It is mathematically defined in the following.

Problem PMS: Does there exist a schedule $S = (\mathbb{S}_1, \mathbb{S}_2, \ldots, \mathbb{S}_k)$,

$$\text{s.t.:} \quad \max_{1 \leqslant i \leqslant k} \sum_{v \in \mathbb{S}_i} l_v \leqslant c_0, \tag{25}$$

$$\bigcup_{i=1}^{k} \mathbb{S}_i = \mathbb{T}, \tag{26}$$

$$\mathbb{S}_i \cap \mathbb{S}_{i'} = \emptyset,\ \forall i, i' = 1,2,\ldots,k,\ i \neq i', \tag{27}$$

where $\mathbb{S}_i$ and $\mathbb{T}$ are the sets of tasks scheduled onto the i^{th} processor and of all tasks respectively.

Then, the complexity result of the LBAP problem is given in the following theorem.

Theorem 3.1: When $D \geqslant 2$, the LBAP problem defined in this paper is NP-hard.

Proof: The basic idea to show this result is to reduce an NP-hard problem to the LBAP problem. The problem we choose is the Parallel Machine Scheduling (PMS) problem, which is NP-hard when the number of parallel machines $k \geqslant 2$ [18].

It is clear that the basic units of the instance of the PMS problem are tasks. Thus, from the PMS problem, we can specify its corresponding instance of the LBAP problem as follows:

The general assembly line system where the problem is defined consists of a single reliable machine (reliable implies the machine's Mean Time Between Failures (MTBF) is infinite), N line-side buffers and D drivers. The system is shown in Fig. 4. For every line-side buffer, the part consumption rate,

the delivery quantity, and the round trip time are deterministic. Additionally, the delivery policy is the Reorder Point (RP) policy, and

$$D = k, \tag{28}$$
$$N = n, \tag{29}$$
$$\mathrm{USG}_{1v} = 1,\ v = 1, 2, \ldots, N, \tag{30}$$
$$\mathrm{QTY}_{1v} = 1,\ v = 1, 2, \ldots, N, \tag{31}$$
$$\mathrm{RTT}_{1v} = l_v,\ v = 1, 2, \ldots, N, \tag{32}$$
$$\tau = \frac{1}{2} \min_{1 \leqslant v \leqslant n} l_v, \tag{33}$$
$$\mathrm{TP}_0 = \frac{1}{c_0}, \tag{34}$$
$$\mathrm{RT} = 0, \tag{35}$$

where USG_{1v}, QTY_{1v}, and RTT_{1v} are respectively the average part usage by the job, the average delivery quantity in one trip, and the average round trip time for part delivery for line-side buffer b^L_{1v}, $v = 1, 2, \ldots, N$, and TP_0, τ, and RT are the desired throughput level, the cycle time of the machine, and the reorder threshold in the material handling subsystem respectively, as defined in Section II.

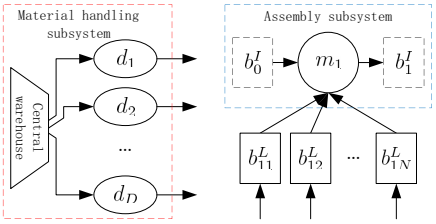


Fig. 4. The line with a single machine

From (28)-(35), it is easy to see that this instance can be constructed in polynomial time. Now, we claim the constructed instance of the LBAP problem has a feasible assignment if and only if there exists a schedule of tasks onto parallel processors with makespan no more than c_0 for the corresponding instance of the PMS problem.

Let's begin with the analytical expression of the throughput of the system shown in Fig. 4 to investigate that the constructed instance of the LBAP problem is a "yes" instance if and only if the corresponding instance of the PMS problem is a "yes" instance.

For the reliable assembly line system shown in Fig. 4, the production process of the system is as follows: as assumed in Subsection II-D, all line-side buffers are empty at the beginning of the production, thus each driver starts delivering parts for the line-side buffers assigned to him/her and the machine has to wait for parts until no line-side buffer is empty. The cycle time of the machine is so short that the machine finishes processing the job before all drivers return to the central warehouse. Obviously, driver d_{i^*}, where $i^* = \arg\max_{1 \leqslant i \leqslant D} \sum_{b^L_{1v} \in \mathbb{A}_i} \mathrm{RTT}_{1v}$, is the latest person finishing part delivery and returning to the central warehouse. According to the Reorder Point (RP) delivery policy, driver d_{i^*} has to deliver parts for the next job as soon as he returns. That is to say, in the process of part delivery, driver d_{i^*} is always busy for part delivery no matter in what order he serves the line-side buffers assigned to him. So, driver d_{i^*} is the bottleneck in the material handling subsystem and thus the bottleneck of the whole system. Therefore, in terms of Definition 2.1, the throughput of the system is

$$\mathrm{TP}^{\pi^{\mathrm{RP}}}(A) = \frac{1}{\max\limits_{1 \leqslant i \leqslant D} \sum\limits_{b^L_{1v} \in \mathbb{A}_i} \mathrm{RTT}_{1v}}. \tag{36}$$

Hence, from (34), the LBAP problem defined in the constructed single-machine system can be presented as follows:

Problem LBAP constructed from PMS: Does there exist an assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$,

$$\text{s.t.:} \quad \max_{1 \leqslant i \leqslant D} \sum_{b^L_{1v} \in \mathbb{A}_i} \mathrm{RTT}_{1v} \leqslant \frac{1}{\mathrm{TP}_0}, \tag{37}$$
$$\bigcup_{i=1}^{D} \mathbb{A}_i = \mathbb{B}, \tag{38}$$
$$\mathbb{A}_i \cap \mathbb{A}_{i'} = \emptyset,\ \forall i, i' = 1, 2, \ldots, D,\ i \neq i'. \tag{39}$$

Any feasible solution of the instance of Problem LBAP constructed from PMS can be used to define a solution to the corresponding instance of Problem PMS: schedule the v^{th} task onto the i^{th} parallel processor if and only if the v^{th} line-side buffer is assigned to the i^{th} driver, $i = 1, 2, \ldots, D$, $v = 1, 2, \ldots, N$. This is obvious because Problem LBAP constructed from PMS is essentially the same as Problem PMS. Thus, if and only if the constructed instance of the LBAP problem has a line-side buffer assignment such that $\mathrm{TP}^{\pi^{\mathrm{RP}}}(A) \geqslant \mathrm{TP}_0$, the instance of the original PMS problem has a schedule under which the latest completion time of all tasks is no later than c_0, which completes the proof. ■

Clearly, from Theorem 3.1, the PMS problem is a special case of the LBAP problem. However, it should be pointed out that the LBAP problem is harder because in general, the analytical expression of the throughput is hard to obtain (if not totally impossible) due to the complex system dynamics and the existence of randomness in the system.

IV. The Solution Methodology for LBAP

As demonstrated in Section III and Subsection II-D, the Line-side Buffer Assignment Problem (LBAP) is hard to solve because of its NP-hardness and the difficulty in efficiently evaluating the system throughput. In this section, we will develop an algorithm for the problem. First, in Subsection IV-A, some definitions will be given to facilitate the algorithm development. Then, in Subsection IV-B, a necessary condition for the LBAP problem having feasible assignments will be developed. After that, we will design the algorithm towards the problem in Subsection IV-C. Finally, the algorithm will be illustrated by a toy line in Subsection IV-D.

A. Definitions

Definition 4.1: A partial assignment A is a fixed partition $(\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$ of some line-side buffers in set $\mathbb{B}$ into D disjoint sets, one for each driver. Mathematically, for partial

assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, $\mathbb{A}_i \cap \mathbb{A}_{i'} = \emptyset$, $\forall i, i' = 1, 2, \ldots, D$, $i \neq i'$, and $\bigcup_{i=1}^{D} \mathbb{A}_i \subseteq \mathbb{B}$, where $\mathbb{B}$ is the set of all line-side buffers.

Remark 4.1: From Definitions 2.3 and 4.1, we know an assignment is a special partial assignment. In this paper, the system performance under a partial assignment is evaluated under the assumption that all line-side buffers not included in the partial assignment have sufficient part supply.

Definition 4.2: Workload of a line-side buffer is the average service time for the line-side buffer per finished product in the steady state of the system.

When the system is in the steady state, in a period of time T, the expected total service time for line-side buffer $b_{uv}^L \in \mathbb{B}$ is $W_{uv} \cdot \mathrm{RTT}_{uv}$, where W_{uv} is the expected total number of trips for b_{uv}^L in the period of time T. Since the system is in the steady state, $W_{uv} \cdot \mathrm{QTY}_{uv} = \mathrm{TP}^{\pi}(A) \cdot T \cdot \mathrm{USG}_{uv}$ holds. Thus, the total service time for line-side buffer b_{uv}^L in the period of time T is $\mathrm{TP}^{\pi}(A) \cdot T \cdot \frac{\mathrm{USG}_{uv} \cdot \mathrm{RTT}_{uv}}{\mathrm{QTY}_{uv}}$. In terms of Definition 4.2, the workload of line-side buffer b_{uv}^L, denoted by ω_{uv}, can be derived as follows:

$$\omega_{uv} = \frac{\mathrm{USG}_{uv} \cdot \mathrm{RTT}_{uv}}{\mathrm{QTY}_{uv}}, \ b_{uv}^L \in \mathbb{B}. \tag{40}$$

Definition 4.3: Driver's workload is his average service time for part delivery per finished product in the steady state of the system.

In terms of Definition 4.3, driver d_i's workload, denoted by $\Omega_i(A)$, can be easily derived as

$$\Omega_i(A) = \sum_{b_{uv}^L \in \mathbb{A}_i} \frac{\mathrm{USG}_{uv} \cdot \mathrm{RTT}_{uv}}{\mathrm{QTY}_{uv}}, i = 1, 2, \ldots, D. \tag{41}$$

Clearly, as with the round trip time, the workload of the line-side buffer and driver's workload are measured in time unit, such as in minutes.

B. Necessary Condition for Feasible Assignments

In the system of general assembly line with material handling, all assignments may be infeasible if there are too few drivers delivering parts for the line-side buffers. In this case, the number of drivers D is infeasible. Since there is no efficient way to identify D_{lean}, the lean number of drivers needed in the material handling subsystem, to possibly avoid solving the LBAP problem with infeasible D, D_{lean} must be estimated so that it is no need to solve the problem with $0 \leqslant D < D_{lean}$.

From Definition 2.2, it is clear that all drivers' utilizations are not beyond 1 under any partial assignment, i.e., $\Gamma_i^{\pi}(A) \leqslant 1, \ \forall A$; from Definitions 2.2 and 4.3, it is easy to obtain $\Gamma_i^{\pi}(A) = \mathrm{TP}^{\pi}(A) \cdot \Omega_i(A)$, $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, $i = 1, 2, \ldots, D$. Thus, we have

$$\mathrm{TP}^{\pi}(A) \cdot \Omega_i(A) \leqslant 1, \ i = 1, 2, \ldots, D. \tag{42}$$

That is to say, no matter how many drivers are in the material handling subsystem and whatever the partial assignment A is, all drivers' utilizations are not beyond 1. Thus, (42) can be viewed as an implicit constraint in the LBAP problem. Combining (42) and (3) and considering (40) and (41), we derive

$$\mathrm{TP}_0 \cdot \sum_{b_{uv}^L \in \mathbb{A}_i} \omega_{uv} \leqslant 1, \ \forall i = 1, 2, \ldots, D. \tag{43}$$

It is obvious that, from (43), $\mathrm{TP}_0 \cdot \max_{1 \leqslant i \leqslant D} \sum_{b_{uv}^L \in \mathbb{A}_i} \omega_{uv} \leqslant 1$ is a necessary condition of the LBAP problem defined in Subsection II-C and the optimal value $\widehat{D}_{lean}^*$ of the following optimization problem is a lower bound of D_{lean}:

Problem of relaxed LBAP:

$$\min \quad D \tag{44}$$

$$\text{s.t.:} \quad \max_{1 \leqslant i \leqslant D} \sum_{b_{uv}^L \in \mathbb{A}_i} \omega_{uv} \leqslant \frac{1}{\mathrm{TP}_0}, \tag{45}$$

$$\bigcup_{i=1}^{D} \mathbb{A}_i = \mathbb{B}, \tag{46}$$

$$\mathbb{A}_i \cap \mathbb{A}_{i'} = \emptyset, \ \forall i, i' = 1, 2, \ldots, D, \ i \neq i'. \tag{47}$$

Clearly, the above problem is an optimization version of the PMS problem, and it is hard to solve in polynomial time because of its NP-completeness. However, we can provide another lower bound which can be easily calculated. Adding (43) with respect to i, we get $\mathrm{TP}_0 \cdot \sum_{b_{uv}^L \in \mathbb{B}} \omega_{uv} \leqslant D$. Thus, another lower bound of D_{lean}, denoted by $\widehat{D}_{lean}$, is

$$\widehat{D}_{lean} = \left\lceil \mathrm{TP}_0 \cdot \sum_{b_{uv}^L \in \mathbb{B}} \omega_{uv} \right\rceil. \tag{48}$$

It should be pointed out that $\widehat{D}_{lean} \leqslant \widehat{D}_{lean}^*$ and $\widehat{D}_{lean}^*$ is a tighter lower bound of D_{lean}. However, (48) is preferable in practice because $\widehat{D}_{lean}$ can be easily calculated and in many cases, it is equal or much close to $\widehat{D}_{lean}^*$. Hence, for a fixed D, $D \geqslant \widehat{D}_{lean}$ is a necessary condition for the LBAP problem having feasible solutions.

C. Algorithm Design

Here we will present a two-phase algorithm to our LBAP problem. In the algorithm, $A_0 = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$ is called the root partial assignment, where $\mathbb{A}_i = \emptyset$, $i = 1, 2, \ldots, D$; $\mathscr{A}_0$ is the set of all assignments. The flowchart of the algorithm is shown in Fig. 9, in which the two dashed rectangles indicate Phase one and Phase two of the algorithm.

As we pointed out in the introduction, our algorithm design is based on two key points: the first is the efficient search of feasible assignments inspired by the Parallel Machine Scheduling (PMS) algorithms; the second, inspired by the Nested Partitions Algorithm [19] [20], is the use of backtracking given the difficulties caused by NP-hardness of the problem, the complex system dynamics, and the evaluation noises. Accordingly, our algorithm structure has two phases.

In the first phase, we take advantages of the analogousness between the LBAP problem and the PMS problem (will be demonstrated in Subsection V-A) to borrow the idea of PMS algorithms to solve the LBAP problem. To be specific, in order to design the first phase of our algorithm, from the Longest Processing Time (LPT) algorithm for the PMS problem, we

adopt the idea of sorting all tasks in decreasing order of their processing times and balancing total processing times on the processors by assigning the sorted tasks one by one. It should be pointed out that in the course of assigning line-side buffers to drivers, constraint (3) must be checked (by simulation) under the partial assignment and observed infeasible partial assignments will be very unlikely to be further considered due to the monotonicity property of the throughput which will be developed in Subsection V-B. The observed feasible partial assignment that balance the driver's workload will be chosen with high probability in each iteration (In case of a tie, we randomly choose any observed feasible partial assignment).

If the first phase of the algorithm arrives at an assignment the feasibility probability (defined in (51)) of which is not high enough or all child partial assignments of some partial assignment are observed to be infeasible, we will start the second phase.

In the second phase, backtracking is introduced in the search procedure. We adopt the Nested Partitions Algorithm idea of exploiting promising subregions (i.e., feasible child partial assignments) if some subregion is promising and backtracking to the superregion (i.e., the parent partial assignment) if not [19] [20]. As usual, we will introduce randomness to help the algorithm to avoid being trapped in a neighborhood of an infeasible assignment while the problem is feasible. In our algorithm, the root partial assignment is regarded as its parent partial assignment and an assignment is regarded as its only child partial assignment.

The algorithm is described in the following.

Algorithm Sequential Assignment with Backtracking (SAB) algorithm

Initialization:

1: Calculate the workload ω_{uv} of line-side buffer $b^L_{uv} \in \mathbb{B}$ using (40) and define the set of unassigned line-side buffers $\mathbb{U} = \{b^L_{(1)}, b^L_{(2)}, \ldots, b^L_{(N)}\}$, which is a list of all line-side buffers sorted in decreasing order of their workloads, where $N = \sum_{u=1}^{M} N^L_u$ is the total number of line-side buffers. Set TP_0 and calculate the lower bound $\widehat{D}_{lean}$ of the lean number of drivers using (48). If the given number of drivers $D < \widehat{D}_{lean}$, report that no feasible assignment exists for D and stop; otherwise go to Step 2.

2: Initialize the current partial assignment $A = A_0$ and drivers' workloads $\Omega_i(A) = 0$, $i = 1, 2, \ldots, D$. Initialize $n_{\widetilde{A}} = 0$, $\overline{\mathrm{TP}}^{\pi}_{n_{\widetilde{A}}}(\widetilde{A}) = 0$, and $S^{\pi}_{n_{\widetilde{A}}}(\widetilde{A}) = 0$, $\forall \widetilde{A} \in \mathscr{A}_0$. Set a small positive value ϵ, a desired probability P^* that observed feasible assignments are true feasible, and a sufficiently large n as the maximum number of random transitions in Phase two. Go to Step 3.

Phase one (straightforwardly assigning):

3: Try to assign the first buffer $b^L_{(j)}$ in $\mathbb{U}$ to driver d_i based on the current partial assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, evaluate the throughput under the new child partial assignment $A_i = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}'_i, \ldots, \mathbb{A}_D)$ of A by simulation for $r > 1$ replications, and obtain samples $\widehat{\mathrm{TP}}^{\pi}_k(A_i)$, $k = 1, 2, \ldots, r$, and the sample mean of the throughput $\overline{\mathrm{TP}}^{\pi}(A_i)$, where $\mathbb{A}'_i = \mathbb{A}_i \cup \{b^L_{(j)}\}$. Accordingly, A will be called as the parent of A_i. If $\overline{\mathrm{TP}}^{\pi}(A_i) < \mathrm{TP}_0$ for $i = 1, 2, \ldots, D$, go to Step 5; otherwise go to Step 4.

4: Remove the first $b^L_{(j)}$ from $\mathbb{U}$. Let $i' = \arg\min_{i \in I} \Omega_i(A)$, where I is the set of indices of all observed feasible partial assignments A_i. Calculate $\Omega_{i'}(A_{i'}) = \Omega_{i'}(A) + \omega_{(j)}$ and replace A by $A_{i'}$. If $\mathbb{U}$ is not empty, go to Step 3, otherwise increase n_A by 1, calculate $\overline{\mathrm{TP}}^{\pi}_{n_A}(A)$, $S^{\pi}_{n_A}(A)$, and P_A based on (49), (50), and (51), and go to Step 6 if $P_A \geqslant P^*$ or go to Step 5 if not.

Phase two (randomly assigning with backtracking):

5: Evaluate the throughput under each child A_i (or under A if it is an assignment) of the current partial assignment A by simulation for $r > 1$ replications and obtain the sample mean $\overline{\mathrm{TP}}^{\pi}(A_i)$ (or $\overline{\mathrm{TP}}^{\pi}(A)$) of the throughput. If $\overline{\mathrm{TP}}^{\pi}(A_i) < \mathrm{TP}_0$, $\forall i \in \{1, 2, \ldots, D\}$, (correspondingly, $\overline{\mathrm{TP}}^{\pi}(A) < \mathrm{TP}_0$), let the backtracking probability $P_b = 1 - \epsilon$, otherwise $P_b = \epsilon$. Generate a random number ζ following UNIFORM$(0, 1)$. Provided that A is not an assignment, i.e., $A \notin \mathscr{A}_0$. If there exists some $i' \in \{1, 2, \ldots, D\}$ such that $\frac{(i'-1)(1-P_b)}{D} < \zeta \leqslant \frac{i'(1-P_b)}{D}$ and $\overline{\mathrm{TP}}^{\pi}(A_{i'}) \geqslant \mathrm{TP}_0$, replace A by $A_{i'}$; or else replace A by its parent partial assignment A'. Provided that A is an assignment, i.e., $A \in \mathscr{A}_0$. If $\zeta \leqslant 1 - P_b$ and $\overline{\mathrm{TP}}^{\pi}(A) \geqslant \mathrm{TP}_0$, keep A as the current partial assignment; or else replace A by its parent A'. Adjust list $\mathbb{U}$ accordingly. If $\mathbb{U}$ is empty, increase n_A by 1, calculate $\overline{\mathrm{TP}}^{\pi}_{n_A}(A)$, $S^{\pi}_{n_A}(A)$, and P_A based on (49), (50), and (51). Decrease n by 1. If $P_A \geqslant P^*$ or $n = 0$, go to Step 6; otherwise, go to Step 5.

Output:

6: If $\max_{A \in \mathscr{A}_0} n_A > 0$, denote $\widehat{A}^* = \arg\max_{A \in \mathscr{A}_0} n_A$ and output the assignment $\widehat{A}^*$ if $P_{\widehat{A}^*} \geqslant P^*$. Otherwise report that no feasible assignment is found for the given D drivers. Stop.

In Step 2 of the algorithm, $n_{\widetilde{A}}$, $\overline{\mathrm{TP}}^{\pi}_{n_{\widetilde{A}}}(A)$, and $S^{\pi}_{n_{\widetilde{A}}}(\widetilde{A})$ are respectively the accumulated visited times of the assignment $\widetilde{A} \in \mathscr{A}_0$, the sample mean and the sample deviation of all samples the simulation model evaluates under $\widetilde{A}$. In addition, $\overline{\mathrm{TP}}^{\pi}_{n_{\widetilde{A}}}(A)$ and $S^{\pi}_{n_{\widetilde{A}}}(\widetilde{A})$ are iteratively updated as follows:

$$\overline{\mathrm{TP}}^{\pi}_{n_{\widetilde{A}}}(\widetilde{A}) = (1 - \frac{1}{n_{\widetilde{A}}})\overline{\mathrm{TP}}^{\pi}_{n_{\widetilde{A}}-1}(\widetilde{A}) + \frac{1}{n_{\widetilde{A}}}\overline{\mathrm{TP}}^{\pi}(\widetilde{A}), \quad (49)$$

$$\begin{aligned}(S^{\pi}_{n_{\widetilde{A}}}(\widetilde{A}))^2 = \frac{1}{n_{\widetilde{A}} \cdot r - 1}\{r(1 - \frac{1}{n_{\widetilde{A}}})(\overline{\mathrm{TP}}^{\pi}(\widetilde{A}) \\ - \overline{\mathrm{TP}}^{\pi}_{n_{\widetilde{A}}-1}(\widetilde{A}))^2 + (r-1)(S^{\pi}(\widetilde{A}))^2 \\ + [(n_{\widetilde{A}} - 1)r - 1](S^{\pi}_{n_{\widetilde{A}}-1}(\widetilde{A}))^2\},\end{aligned} \quad (50)$$

where $\overline{\mathrm{TP}}^{\pi}(\widetilde{A})$ and $S^{\pi}(\widetilde{A})$ are the sample mean and the sample deviation of the throughput obtained by the new additional evaluation.

Based on (49) and (50), we can calculate the feasibility probability $P_{\widetilde{A}}$ of the assignment $\widetilde{A} \in \mathscr{A}_0$ in Steps 4 and 5 of the algorithm as follows:

$$P_{\widetilde{A}} = F_t(\frac{\sqrt{n_{\widetilde{A}} \cdot r}(\overline{\mathrm{TP}}^{\pi}_{n_{\widetilde{A}}}(\widetilde{A}) - \mathrm{TP}_0)}{S^{\pi}_{n_{\widetilde{A}}}(\widetilde{A})}, n_{\widetilde{A}} \cdot r - 1), \quad (51)$$

where $F_t(\cdot,\cdot)$ is the cumulative distribution function of t-distribution.

Remark 4.2: In the algorithm, ϵ is introduced so that the algorithm can visit all assignments with positive probability. If ϵ is too small, the algorithm may be trapped in a neighborhood of an infeasible assignment; if it is too large, it will take a long time for the algorithm to visit an assignment in Phase two. Thus, $\epsilon \in (\frac{1}{10(D+1)}, \frac{1}{D+1})$ would be appropriate, where D is the number of drivers.

Remark 4.3: As will be shown in Section V, the first phase of the algorithm helps to effectively search feasible partial assignments by taking advantage of the problem specific knowledge. General random search algorithms without using similar knowledge would be much less effective as the example will be given in Subsection IV-D.

Remark 4.4: The concept of Nested Partitions Algorithm is used in the second phase of the algorithm. Clearly, it can be replaced by other random search algorithms. However, there is generally no guarantee for these algorithms to converge to feasible assignments with probability 1.

D. An Example Line

We illustrate the Sequential Assignment with Backtracking (SAB) algorithm with a toy line, which consists of two machines, one in-process buffer (excluding the input and output in-process buffers), and five line-side buffers. The line is shown in Fig. 5. The number of drivers is $D = 2$. Parameters of machines, the in-process buffer, and line-side buffers are displayed in Tables I-III. We set the desired throughput TP_0 as $0.957\mathrm{TP}_{\max}$, where $\mathrm{TP}_{\max}$ is the ideal throughput when all line-side buffers have sufficient part supply. The blocking mechanism of machines is Blocking-After-Service. Failure times and repair times of machines are independently and exponentially distributed, and drivers follow the Reorder Point (RP) delivery policy and RT is 15 minutes. The consumption number of parts by each job in line-side buffer $b^L_{uv} \in \mathbb{B}$ follows two-point distribution, in which alternative values are $\lfloor \mathrm{USG}_{uv} \rfloor$ and $\lceil \mathrm{USG}_{uv} \rceil$ with mean USG_{uv}, and round trip time for part delivery and delivery quantity for b^L_{uv} in each trip are invariably RTT_{uv} and QTY_{uv} respectively. In the algorithm, the system throughput is evaluated by a simulation model established based on the method introduced in [5]. $10,000$ minutes (about 21 subsequent shifts) including $5,000$ minutes warm-up time of the production is simulated and the simulation is run for 20 replications. The total number of finished products arriving at the output in-process buffer after the warm-up time is collected.

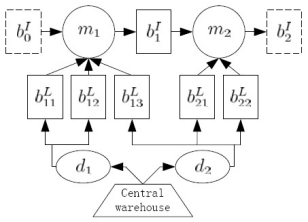


Fig. 5. The example line

TABLE I
PARAMETERS OF MACHINES

m_u	MTBF (min)	MTTR (min)	τ_u (min)
m_1	100	0.5	1
m_2	50	0.4	1.2

TABLE II
PARAMETERS OF IN-PROCESS BUFFERS

b^I_u	C^I_u
b^I_1	2

TABLE III
PARAMETERS OF LINE-SIDE BUFFERS

b^L_{uv}	USG_{uv}	QTY_{uv}	RTT_{uv} (min)	C^L_{uv}
b^L_{11}	0.75	18	18	100
b^L_{12}	0.27	7	8.4	100
b^L_{13}	0.6	20	18	100
b^L_{21}	0.1	8	10	100
b^L_{22}	0.17	4	10	100

The process of the algorithm is as follows. The workloads of the line-side buffers are calculated in terms of (40) and shown in Table IV. The total workload of the line-side buffers is $\sum_{b^L_{uv}\in\mathbb{B}} \omega_{uv} = 2.164$ (min) and $\mathrm{TP}_{\max}$ is 49.61 jobs per hour (jph). Thus, the desired throughput level is $\mathrm{TP}_0 = 0.957\mathrm{TP}_{\max} = 47.48$ (jph). Then, we calculate the lower bound of the lean number of drivers $\widehat{D}_{lean}$ using (48), and have $\mathrm{TP}_0 \cdot \sum_{b^L_{uv}\in\mathbb{B}} \omega_{uv} = 1.71$ and $\widehat{D}_{lean} = 2$. Since $D \geqslant \widehat{D}_{lean}$, we set $\epsilon = 0.05$, $P^* = 0.95$, and $n = 20$, and proceed the algorithm to the first phase of straightforward search for a feasible assignment. Stages and results of the assignment process in Phase one are summarized in Table IV, in which line-side buffers are listed in decreasing order of their workloads. The process of workload balancing is shown in Fig. 6, in which the length of the block represents the workload of corresponding line-side buffer. In Phase one, the algorithm finds an assignment $A = (\mathbb{A}_1, \mathbb{A}_2)$, where $\mathbb{A}_1 = \{b^L_{11}, b^L_{12}\}$ and $\mathbb{A}_2 = \{b^L_{13}, b^L_{21}, b^L_{22}\}$. Since the feasibility probability of this assignment is $0.972 > P^*$, the algorithm will not proceed to Phase two and stop. Thus, $\widehat{A}^* = (\mathbb{A}_1, \mathbb{A}_2)$, where $\mathbb{A}_1 = \{b^L_{11}, b^L_{12}\}$ and $\mathbb{A}_2 = \{b^L_{13}, b^L_{21}, b^L_{22}\}$. Under this assignment, the estimated system throughput is 47.51 (jph) and drivers' utilizations are 0.86 and 0.85. If the samples of the throughput are identically and independently distributed with a normal distribution, the 95% confidence interval is $(47.33, 47.70)$. The assignment is shown in Fig. 5.

TABLE IV
THE PROCESS OF THE LINE-SIDE BUFFER ASSIGNMENT IN PHASE ONE

b^L_{uv}	ω_{uv} [1]	A	$\mathrm{TP}^{\pi^{\mathrm{RP}}}$ [2]
b^L_{11}	0.75	$\mathbb{A}_1 = \{b^L_{11}\}, \mathbb{A}_2 = \emptyset$	49.61
b^L_{13}	0.54	$\mathbb{A}_1 = \{b^L_{11}\}, \mathbb{A}_2 = \{b^L_{13}\}$	49.61
b^L_{22}	0.425	$\mathbb{A}_1 = \{b^L_{11}\}, \mathbb{A}_2 = \{b^L_{13}, b^L_{22}\}$	48.80
b^L_{12}	0.324	$\mathbb{A}_1 = \{b^L_{11}, b^L_{12}\}, \mathbb{A}_2 = \{b^L_{13}, b^L_{22}\}$	48.55
b^L_{21}	0.125	$\mathbb{A}_1 = \{b^L_{11}, b^L_{12}\}, \mathbb{A}_2 = \{b^L_{13}, b^L_{21}, b^L_{22}\}$	47.51

[1] measured by minutes
[2] measured by jobs per hour (jph)

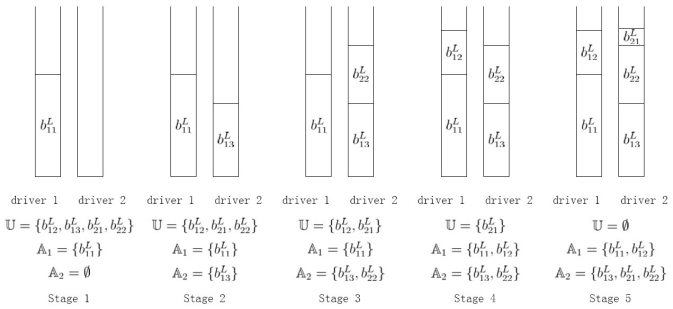


Fig. 6. The process of workload balancing in Phase one

To demonstrate the effectiveness of the SAB algorithm, Genetic Algorithms (GA) is also implemented as a comparison. The population size is set to be 6 (total number of assignments is 15) and crossover and mutation probabilities are respectively 0.3 and 0.1. If the algorithm finds an assignment with feasibility probability larger than P^*, it stops and we regard it finds a feasible assignment. Considering there are at most 5 iterations in the first phase of the SAB algorithm, to fairly compare the algorithms, we set the maximum number of generations of GA to be the maximum number of iterations in the second phase of the SAB algorithm plus 5. Since both algorithms are random algorithms, the experiments are repeated for 100 times. Clearly, sometimes the algorithms can find a feasible assignment and sometimes cannot. Total number of times that the algorithm fails to find the feasible assignments is counted and shown in Table V. Obviously, the SAB algorithm outperforms GA on this toy example.

TABLE V
COMPARISON WITH GENETIC ALGORITHMS

Maximum #iteration	Total #failures	
	SAB	GA
20	16	25
50	12	11
100	4	11

V. ANALYSIS OF THE ALGORITHM

In this section, we will investigate the two structural characteristics of the Line-side Buffer Assignment Problem (LBAP) employed in the Sequential Assignment with Backtracking (SAB) algorithm and establish the global convergence of the algorithm. First, we will investigate the analogousness between the LBAP problem and the Parallel Machine Scheduling (PMS) problem in Subsection V-A. Then, the monotonicity of the system throughput with respect to the partial assignment in the course of assigning line-side buffers to drivers will be exploited in Subsection V-B. Finally, the global convergence of the SAB algorithm is established in Subsection V-C.

A. Analogousness Between LBAP and PMS

The analogousness between the LBAP problem and the PMS problem is the basis of the SAB algorithm presented in Subsection IV-C. In this subsection, we will investigate the analogousness between these two problems, which is implied in Section III.

In the single-machine system in Fig. 4, it is clear that drivers and line-side buffers can be respectively regarded as "parallel processors" and "tasks" in the PMS point of view, and RTT (average round trip time) as the "processing time" of the "task". However, in a generalized system of general assembly line with material handling, RTT cannot be regarded as the "processing time" anymore because in general, frequencies of driver's part delivery are not identical for all line-side buffers.

Nevertheless, the workload of the line-side buffer defined in Definition 4.2 can be regraded, in the PMS point of view, as the "processing time" instead in a generalized system of general assembly line with material handling. In fact, from (40) and (41), we can easily derive the following equation:

$$\Omega_i(A) = \sum_{b_{uv}^L \in \mathbb{A}_i} \omega_{uv}, i = 1, 2, \ldots, D. \tag{52}$$

Equation (52) implies in the LBAP problem, drivers and line-side buffers can still be viewed as "parallel processors" and "tasks" in the point of view of PMS, while the workload ω of the line-side buffer other than RTT is the "processing time" of the "task" on the "processor". That is to say, the LBAP problem is analogous to the PMS problem. Actually, (40) indicates that ω is the weighted RTT, where the weight is $\frac{\mathrm{USG}}{\mathrm{QTY}}$, the average times of part delivery for the line-side buffer per finished product. In the single-machine system in Fig. 4, $\frac{\mathrm{USG}}{\mathrm{QTY}} = 1$ for all line-side buffers, which is why RTT can be regarded as the "processing time" in PMS point of view in Section III.

Although the LBAP problem is harder than the PMS problem in two aspects–the PMS problem is a special case of the LBAP problem and the feasibility checking of the LBAP problem is time-consuming, it is natural to borrow the concepts of PMS algorithms to solve our problem. The Longest Processing Time (LPT) algorithm, which is proposed and first analyzed in [22] and [23] respectively, is well-known and effective for solving the PMS problem. The essence of the LPT algorithm–sorting all tasks in decreasing order of their processing times and balancing total processing times on the processors by assigning the sorted tasks one by one–make it an asymptotically optimal algorithm for the PMS problem [24]. Thus, as with the LPT algorithm, the SAB algorithm for the LBAP problem sorts all line-side buffers in decreasing order of their workloads and assigns them to drivers one by one to balance drivers' workloads as long as there is an opportunity to refine a partial assignment. However, different from the PMS problem, the system dynamics has complex impact on the throughput. Thus, the rationality of balancing drivers' workloads to search a feasible assignment should be validated. The rationality will be demonstrated by numerical experiments in Subsection VI-A.

It is worth noting that, although the Bin-Packing Problem (BPP) is a dual problem to the PMS problem, the BPP algorithms are not suitable to solve the LBAP problem. Clearly, in the point of view of BPP, drivers and line-side buffers are regraded as "bins" and "items" respectively. To use the BPP algorithms, "bin capacities" must be determined, which is hard in a generalized system of general assembly line with material handling. For the single-machine system defined in

Section III, "bin capacities" are $\frac{1}{\mathrm{TP}_0}$ because of (36) indicating $\mathrm{TP}^{\pi}(A) \cdot \max_{1 \leqslant i \leqslant D} \Omega_i = 1$, which implies the bottleneck of the system is the driver of maximum workload in the material handling subsystem, no matter what the assignment is. However, for a generalized system, it is not always the case. Thus it cannot be taken for granted that "bin capacities" are $\frac{1}{\mathrm{TP}_0}$. Therefore, BPP algorithms, such as Best Fit Decreasing (BFD) and First Fit Decreasing (FFD) algorithms, are not suitable to solve the LBAP problem. In addition, the MULTIFIT algorithm [17], another heuristic algorithm for the PMS problem, is also not applicable because it solves the PMS problem by iteratively solving a series of BPPs by the FFD algorithm.

B. Monotonicity Property of the Throughput

In the process of solving the Parallel Machine Scheduling (PMS) problem, as tasks are scheduled onto processors one by one, the latest completion time of tasks on the processors is non-decreasing. Based on the analogousness between the LBAP problem and the PMS problem, the throughput of the system of general assembly line with material handling would be very likely to be non-increasing because, in the single-machine system in Section III, it is the reciprocal of the maximum of drivers' workload. Thus, in this subsection, we will investigate the monotonicity of the system throughput in the course of assigning line-side buffers to drivers.

Let $\widetilde{T}_{uv}^{pr}(q)$ be an alternative of $T_{uv}^{pr}(q)$ under the same realization of uncertainties of the system, where $\sum_{j=1}^{k-1} Q_{j,uv}^{C} < q \leqslant \sum_{j=1}^{k} Q_{j,uv}^{C}$, $b_{uv}^{L} \in \mathbb{B}$, $k = 1, 2, \ldots$, and the throughput TP and $\widetilde{\mathrm{TP}}$ and all other timings are denoted accordingly. Then we have following propositions and theorems.

Proposition 5.1: In the material handling subsystem, for all line-side buffers $b_{uv}^{L} \in \mathbb{B}$, if the release times of all parts are not delayed, i.e., $\widetilde{T}_{uv}^{pr}(q) \leqslant T_{uv}^{pr}(q)$, $\sum_{j=1}^{k-1} Q_{j,uv}^{C} < q \leqslant \sum_{j=1}^{k} Q_{j,uv}^{C}$, $k = 1, 2, \ldots$, then the arrival times of all finished products at the output in-process buffer will not be delayed in time interval $[0, T]$, i.e., $\widetilde{T}_{M}^{a}(k) \leqslant T_{M}^{a}(k)$.

Proof: It is proved in Appendix B-A. ∎

Lemma 5.1: In the material handling subsystem, for all line-side buffers $b_{uv}^{L} \in \mathbb{B}$, if the release times of all parts are not delayed, i.e., $\widetilde{T}_{uv}^{pr}(q) \leqslant T_{uv}^{pr}(q)$, $\sum_{j=1}^{k-1} Q_{j,uv}^{C} < q \leqslant \sum_{j=1}^{k} Q_{j,uv}^{C}$, $k = 1, 2, \ldots$, then the throughput will not drop, i.e., $\widetilde{\mathrm{TP}} \geqslant \mathrm{TP}$.

Proof: From Proposition 5.1, we know that the arrival times of finished products at the output in-process buffer in time interval $[0, T]$ are not delayed. Therefore, the number of finished products arriving at the output in-process buffer in $[0, T]$ will not drop. Let $K_T(\xi)$ and $\widetilde{K}_T(\xi)$ denote the numbers of finished products produced in $[0, T]$ accordingly, then we have $K_T(\xi) \leqslant \widetilde{K}_T(\xi)$. From Definition 2.1, we have

$$\mathrm{TP} = \lim_{T \to \infty} \frac{E_{\xi}[K_T(\xi)]}{T} \leqslant \lim_{T \to \infty} \frac{E_{\xi}[\widetilde{K}_T(\xi)]}{T} = \widetilde{\mathrm{TP}}, \quad (53)$$

which completes the proof. ∎

Proposition 5.2: In the material handling subsystem, under the partial assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, drivers follow a delivery policy π to deliver parts for the line-side buffers. For another partial assignment $\widetilde{A} = (\widetilde{\mathbb{A}}_1, \widetilde{\mathbb{A}}_2, \ldots, \widetilde{\mathbb{A}}_D)$, $\widetilde{\mathbb{A}}_{i_0} = \mathbb{A}_{i_0} \setminus \{b_{u_0 v_0}^{L}\}$ for some i_0, where $b_{u_0 v_0}^{L} \in \mathbb{A}_{i_0}$, and $\widetilde{\mathbb{A}}_{i'} = \mathbb{A}_{i'}$, $\forall i' = 1, 2, \ldots, D$, $i' \neq i_0$, $\exists$ a delivery policy $\widetilde{\pi}$, under which for all line-side buffers $b_{uv}^{L} \in \mathbb{B}$, the release times of all parts will not be delayed, i.e., $\widetilde{T}_{uv}^{pr}(q) \leqslant T_{uv}^{pr}(q)$, $q = 1, 2, \ldots$

Proof: It is proved in Appendix B-B. ∎

Lemma 5.2: In the material handling subsystem, under the partial assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, drivers follow a delivery policy π to deliver parts for the line-side buffers. For another partial assignment $\widetilde{A} = (\widetilde{\mathbb{A}}_1, \widetilde{\mathbb{A}}_2, \ldots, \widetilde{\mathbb{A}}_D)$, $\widetilde{\mathbb{A}}_{i_0} = \mathbb{A}_{i_0} \setminus \{b_{u_0 v_0}^{L}\}$ for some i_0, where $b_{u_0 v_0}^{L} \in \mathbb{A}_{i_0}$, and $\widetilde{\mathbb{A}}_{i'} = \mathbb{A}_{i'}$, $\forall i' = 1, 2, \ldots, D$, $i' \neq i_0$, $\exists$ a delivery policy $\widetilde{\pi}$ such that $\mathrm{TP}^{\pi}(A) \leqslant \mathrm{TP}^{\widetilde{\pi}}(\widetilde{A})$.

Proof: From Propositions 5.2 and 5.1 and Lemma 5.1, it is easy to draw the conclusion. ∎

Assumption 5.1: There exists an optimal delivery policy π^* for the part delivery, i.e., under any partial assignment A, $\forall \pi$, $\mathrm{TP}^{\pi^*}(A) \geqslant \mathrm{TP}^{\pi}(A)$ holds.

Theorem 5.1 (Monotonicity theorem 1): Assume Assumption 5.1 holds. In the material handling subsystem, under the partial assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, drivers follow an optimal delivery policy π^* to deliver parts for the line-side buffers. For another partial assignment $\widetilde{A} = (\widetilde{\mathbb{A}}_1, \widetilde{\mathbb{A}}_2, \ldots, \widetilde{\mathbb{A}}_D)$, $\widetilde{\mathbb{A}}_{i_0} = \mathbb{A}_{i_0} \setminus \{b_{u_0 v_0}^{L}\}$ for some i_0, where $b_{u_0 v_0}^{L} \in \mathbb{A}_{i_0}$, and $\widetilde{\mathbb{A}}_{i'} = \mathbb{A}_{i'}$, $\forall i' = 1, 2, \ldots, D$, $i' \neq i_0$, the throughput will not drop, i.e., $\mathrm{TP}^{\pi^*}(A) \leqslant \mathrm{TP}^{\pi^*}(\widetilde{A})$.

Proof: In terms of Lemma 5.2, there exists a feasible delivery policy $\widetilde{\pi}$ satisfying $\mathrm{TP}^{\pi^*}(A) \leqslant \mathrm{TP}^{\widetilde{\pi}}(\widetilde{A})$. Due to π^* is an optimal delivery policy, we have $\mathrm{TP}^{\widetilde{\pi}}(\widetilde{A}) \leqslant \mathrm{TP}^{\pi^*}(\widetilde{A})$. Thus, $\mathrm{TP}^{\pi^*}(A) \leqslant \mathrm{TP}^{\pi^*}(\widetilde{A})$ holds. ∎

Theorem 5.1 indicates that, under the optimal delivery policy, the throughput will not increase if more additional line-side buffers are assigned to drivers, which implies that if a partial assignment is infeasible, all partial assignments derived by assigning more line-side buffers to the drivers based on the infeasible one remain infeasible. Obviously, the monotonicity property in Theorem 5.1 can largely reduce the search space to improve the search efficiency of the SAB algorithm developed in Subsection IV-C.

Assumption 5.2: Under all partial assignments, the sample means $\overline{\mathrm{TP}}^{\pi}(A)$ of the samples of the system throughput have the same distribution with mean $\mathrm{TP}^{\pi}(A)$ and variance σ^2.

Theorem 5.2 (Monotonicity theorem 2): Assume Assumptions 5.1 and 5.2 hold. In the material handling subsystem, under the partial assignment $A = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, drivers follow an optimal delivery policy π^* to deliver parts for the line-side buffers. For another partial assignment $\widetilde{A} = (\widetilde{\mathbb{A}}_1, \widetilde{\mathbb{A}}_2, \ldots, \widetilde{\mathbb{A}}_D)$, $\widetilde{\mathbb{A}}_{i_0} = \mathbb{A}_{i_0} \setminus \{b_{u_0 v_0}^{L}\}$ for some i_0, where $b_{u_0 v_0}^{L} \in \mathbb{A}_{i_0}$, and $\widetilde{\mathbb{A}}_{i'} = \mathbb{A}_{i'}$, $\forall i' = 1, 2, \ldots, D$, $i' \neq i_0$, the feasibility probability will not drop, i.e., $\mathcal{P}[\overline{\mathrm{TP}}^{\pi^*}(A) \geqslant \mathrm{TP}_0] \leqslant \mathcal{P}[\overline{\mathrm{TP}}^{\pi^*}(\widetilde{A}) \geqslant \mathrm{TP}_0]$.

Proof: Let $F(x)$ be the normalized distribution function of the sample mean of the samples of the system throughput. In terms of Theorem 5.1, $\mathrm{TP}^{\pi^*}(A) \leqslant \mathrm{TP}^{\pi^*}(\widetilde{A})$. Thus we have $\mathcal{P}[\overline{\mathrm{TP}}^{\pi^*}(A) \geqslant \mathrm{TP}_0] = 1 - F(\frac{\mathrm{TP}_0 - \mathrm{TP}^{\pi^*}(A)}{\sigma}) \leqslant 1 - F(\frac{\mathrm{TP}_0 - \mathrm{TP}^{\pi^*}(\widetilde{A})}{\sigma}) = \mathcal{P}[\overline{\mathrm{TP}}^{\pi^*}(\widetilde{A}) \geqslant \mathrm{TP}_0]$, which completes

the proof. ■

In real systems, the delivery policy is usually a simple rule which is easy to implement on the factory floor, such as the Reorder Point (RP) delivery policy π^{RP}. Although perhaps the RP delivery policy π^{RP} is not optimal, it intuitively makes sense so that it is believed the conclusion in Theorem 5.1 also approximately holds. The approximate monotonicity of the system throughput under the RP delivery policy will be demonstrated by numerical experiments in Subsection VI-A.

C. Global Convergence of the SAB Algorithm

Inspired by the proof of global convergence of Nested Partitions Algorithm in [19] and [20], in this subsection, we will establish the global convergence of the Sequential Assignment with Backtracking (SAB) algorithm developed in Subsection IV-C.

Let $A(k)$ denote the partial assignment visited in the k^{th} iteration of Phase two of the SAB algorithm. Thus, the sequence of partial assignments that the SAB algorithm visits is $\{A(k)\}_{k=1}^{n}$. Let $\mathscr{A}$ and $\mathscr{A}_0^*$ respectively denote the set of all possible partial assignments that could be visited by the SAB algorithm and the set of all truly feasible assignments. Recall that $\mathscr{A}_0$ is the set of all assignments and A_0 is the root partial assignment that no line-side buffer is assigned, i.e., $A_0 = (\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_D)$, $\mathbb{A}_i = \emptyset$, $i = 1, 2, \ldots, D$.

Assumption 5.3: The simulation time is long enough so that 1) for all truly feasible partial assignments $A \in \mathscr{A}$, $\mathcal{P}[\overline{\mathrm{TP}}^{\pi}(A) < \mathrm{TP}_0 | \mathrm{TP}^{\pi}(A) \geqslant \mathrm{TP}_0] \leqslant \alpha$; 2) for all truly infeasible partial assignments $A \in \mathscr{A}$, $\mathcal{P}[\overline{\mathrm{TP}}^{\pi}(A) \geqslant \mathrm{TP}_0 | \mathrm{TP}^{\pi}(A) < \mathrm{TP}_0] \leqslant \beta$; 3) $\alpha \leqslant \epsilon$, $\alpha + \beta \leqslant 1$, $1 - \beta(1-\alpha)(1-\epsilon) \leqslant (1-\alpha)^2$, and $\beta \leqslant \epsilon^{N-1}[1 - \beta(\epsilon + \beta - 2\epsilon\beta)]$, where $\alpha, \beta > 0$, ϵ is a small positive value, and N is the total number of line-side buffers.

Remark 5.1: When the simulation time is long enough, α and β, probabilities of the type I and type II errors, can be so close to 0 that $\alpha \leqslant \epsilon$, $\alpha+\beta \leqslant 1$, $1-\beta(1-\alpha)(1-\epsilon) \leqslant (1-\alpha)^2$, and $\beta \leqslant \epsilon^{N-1}[1 - \beta(\epsilon + \beta - 2\epsilon\beta)]$.

Let us derive the transition probabilities according to the SAB algorithm developed in Subsection IV-C. Consider partial assignment A is not an assignment. According to the transitions defined in Step 5 of the SAB algorithm, the one-step transition probability from A to one of its children A_i is $\frac{1}{D}[(1-\epsilon)P_f + \epsilon(1-P_f)]\mathcal{P}[\overline{\mathrm{TP}}^{\pi}(\widetilde{A}) \geqslant \mathrm{TP}_0]$, where P_f is the probability that at least one of $\overline{\mathrm{TP}}^{\pi}(A_i) \geqslant \mathrm{TP}_0$ holds. Thus, we have

$$P(A, \widetilde{A}) = \begin{cases} \frac{1}{D}(\epsilon + P_f - 2\epsilon P_f)\mathcal{P}[\overline{\mathrm{TP}}^{\pi}(\widetilde{A}) \geqslant \mathrm{TP}_0], \\ \qquad \text{if } \widetilde{A} = A_i, \\ 1 - \sum_{i=1}^{D} P(A, A_i), \text{ if } \widetilde{A} = A', \\ 0, \text{ otherwise}, \end{cases} \tag{54}$$

where A is not an assignment, A_i is one of its children, and A' is its parent; similarly,

$$P(A, \widetilde{A}) = \begin{cases} (\epsilon + P_f - 2\epsilon P_f)\mathcal{P}[\overline{\mathrm{TP}}^{\pi}(\widetilde{A}) \geqslant \mathrm{TP}_0], \\ \qquad \text{if } \widetilde{A} = A, \\ 1 - P(A, A), \text{ if } \widetilde{A} = A', \\ 0, \text{ otherwise}, \end{cases} \tag{55}$$

where A is an assignment, A' is its parent partial assignment, and P_f is the probability that $\overline{\mathrm{TP}}^{\pi}(A) \geqslant \mathrm{TP}_0$ holds.

Remark 5.2: Clearly, probabilities in (54) and (55) indicate we encourage more frequent visit to those (partial) assignments that are more likely to be feasible based on simulation. In addition, the one-step transition probabilities from one partial assignment to its parent and child partial assignments in (54) and (55) are in $(0, 1)$. As a consequence, for any partial assignment A, there exists some $\kappa \leqslant N$ such that the κ-step transition probabilities $P^{(\kappa)}(A, A_0) > 0$ and $P^{(\kappa)}(A_0, A) > 0$. This property will be used in the following proposition.

Proposition 5.3: Assume Assumption 5.3 holds. Then the sequence of partial assignments $\{A(k)\}_{k=1}^{\infty}$ is a time reversible Markov chain and it has a unique stationary distribution $\{\nu(A)\}_{A \in \mathscr{A}}$.

Proof: It is proved in Appendix B-C. ■

Let $\widehat{A}_n^*$ denote the assignment provided by the SAB algorithm after n random transitions in Phase two. Then we have the following lemma.

Proposition 5.4 (coming from [25]): The assignment $\widehat{A}_n^*$ converges to an assignment of maximum stationary distribution of the Markov chain, that is

$$\lim_{n \to \infty} \widehat{A}_n^* \in \arg\max_{A \in \mathscr{A}_0} \nu(A), w.p.1. \tag{56}$$

Consider the Markov chain $\{A(k)\}_{k=1}^{\infty}$ generated by the SAB algorithm. The graph of the chain means the graph having all partial assignments as nodes and the transition probabilities of the chain as edge weights. Let $\kappa(A, \widetilde{A}) = \min\{\kappa : P^{(\kappa)}(A, \widetilde{A}) > 0\}$ be the length of the shortest path from partial assignment A to $\widetilde{A}$ on the graph of the chain, $\forall A, \widetilde{A} \in \mathscr{A}$. Clearly, $\kappa(A, \widetilde{A}) = \kappa(\widetilde{A}, A)$, $\forall A, \widetilde{A} \in \mathscr{A}_0$.

Proposition 5.5: Assume Assumptions 5.1, 5.2, and 5.3 hold. For all infeasible assignment $\widetilde{A} \in \mathscr{A}_0 \setminus \mathscr{A}_0^*$ and feasible assignment $A \in \mathscr{A}_0^*$, $P^{(\kappa(\widetilde{A}, A))}(\widetilde{A}, A) \geqslant P^{(\kappa(A, \widetilde{A}))}(A, \widetilde{A})$.

Proof: It is proved in Appendix B-D. ■

Theorem 5.3: Assume the number of drivers $D \geqslant D_{lean}$ and Assumptions 5.1, 5.2, and 5.3 hold. Then the SAB algorithm developed in Subsection IV-C converges with probability 1 to feasible assignments of the LBAP problem as $n \to \infty$.

Proof: From the definition of the learn number of drivers, there always exists a feasible assignment for $D \geqslant D_{lean}$. Let $\widetilde{A}$ be an infeasible partial assignment and A be a feasible assignment. We denote $\widetilde{A}, \widetilde{A}', \widetilde{A}^{(2)}, \ldots, \widetilde{A}^{(\kappa(\widetilde{A}, \bar{A})-1)}, \bar{A}$, $A^{(\kappa(A, \bar{A})-1)}, \ldots, A^{(2)}, A', A$ as the shortest path from $\widetilde{A}$ to A on the graph of the Markov chain generated by the SAB algorithm, where $\bar{A}$ is the closest common ancestor partial assignment of $\widetilde{A}$ and A. From Proposition 5.3, the Markov chain $\{A(k)\}_{k=1}^{\infty}$ is time reversible. Thus, the stationary distribution ν satisfies the extended version of detailed balance condition [25], namely, for any pair of partial assignments $\widetilde{A}$ and A, the following holds:

$$\begin{aligned} &\nu(\widetilde{A})P(\widetilde{A}, \widetilde{A}')P(\widetilde{A}', \widetilde{A}^{(2)}) \cdots P(\widetilde{A}^{(\kappa(\widetilde{A}, \bar{A})-1)}, \bar{A}) \\ &\quad \cdot P(\bar{A}, A^{(\kappa(A, \bar{A})-1)}) \cdots P(A^{(2)}, A')P(A', A) \\ =&\nu(A)P(A, A')P(A', A^{(2)}) \cdots P(A^{(\kappa(A, \bar{A})-1)}, \bar{A}) \\ &\quad \cdot P(\bar{A}, \widetilde{A}^{(\kappa(\widetilde{A}, \bar{A})-1)}) \cdots P(\widetilde{A}^{(2)}, \widetilde{A}')P(\widetilde{A}', \widetilde{A}). \end{aligned} \tag{57}$$

In other words, we have

$$\nu(\widetilde{A})P^{(\kappa(\widetilde{A},A))}(\widetilde{A},A)=\nu(A)P^{(\kappa(A,\widetilde{A}))}(A,\widetilde{A}). \tag{58}$$

From Proposition 5.5, we have $\nu(A)\geqslant\nu(\widetilde{A})$, $\forall\widetilde{A}\in\mathscr{A}_0\setminus\mathscr{A}_0^*, A\in\mathscr{A}_0^*$.

Thus, if $\lim_{n\to\infty}\widehat{A}_n^*=A^*$, then in terms of Proposition 5.4, $A^*\in\arg\max_{A\in\mathscr{A}_0}\nu(A)\ w.p.1$, which means $\nu(A^*)\geqslant\nu(\widetilde{A})$, $\forall\widetilde{A}\in\mathscr{A}_0$. Since there exists a feasible assignment, A^* is a feasible assignment, which completes the proof. ■

Although the SAB algorithm is globally convergent when $D\geqslant D_{lean}$ and $n\to\infty$, it is hard to check whether $D\geqslant D_{lean}$ holds and run the second phase of the algorithm for infinite iterations. Thus, in practice, we check the necessary condition $D\geqslant\widehat{D}_{lean}$ and set a reasonably large n.

VI. Numerical Experiments

In this section, the rationality of the surrogate criterion of balancing drivers' workloads, the approximate monotonicity of the system throughput under the Reorder Point (RP) delivery policy, and the effectiveness of the Sequential Assignment with Backtracking (SAB) algorithm are numerically demonstrated. Specifically, in Subsection VI-A, we exemplify the toy line introduced in Subsection IV-D to demonstrate the rationality of the surrogate criterion and approximate monotonicity of the throughput under the RP delivery policy. And then, in Subsection VI-B, the SAB algorithm is tested on a real line to demonstrate its effectiveness to the LBAP problem.

A. Rationality Demonstration

In this subsection, we exemplify the toy line shown in Fig. 5 to demonstrate the rationality of the surrogate criterion of balancing drivers' workloads to search a feasible assignment and the approximate monotonicity of the system throughput in the course of assigning line-side buffers to drivers under the RP delivery policy π^{RP}.

First, we demonstrate the rationality of the surrogate criterion of balancing drivers' workloads as mentioned in Subsection V-A. We list all alternative assignments with 2 drivers in Table VI. We use the standard deviation of drivers' workloads to denote the degree of workload balancing among drivers and call it Workload Balancing Measure (WBM). WBMs, throughputs of all alternative assignments are listed in Table VI and only the first assignment is observed to be feasible. The assignments are sorted in increasing order of WBM, and it is clear that, in general, the estimated throughput under the assignment with smaller WBM is larger and all observed feasible assignments have small WBM. Thus, it makes sense to balance drivers' workloads to search a feasible assignment in Phase one of the SAB algorithm.

Second, under the RP delivery policy, we demonstrate the approximate monotonicity of the system throughput in the course of assigning line-side buffers to drivers as mentioned in Subsection V-B. From Table IV, we can see that the system throughput approximately monotonically decreases as line-side buffers are assigned to drivers ("approximately" is reflected in stages 1 and 2, where $\mathrm{TP}^{\pi^{\mathrm{RP}}}(A)>\mathrm{TP}_{\max}$. It might be caused either by the evaluation noises of the simulation model or by the RP delivery policy). Thus, the monotonicity of the throughput approximately holds under the RP delivery policy π^{RP}.

TABLE VI
All Alternative Assignments

$A=(\mathbb{A}_1,\mathbb{A}_2)$	WBM (min)	$\mathrm{TP}^{\pi^{\mathrm{RP}}}$ (jph)
$\mathbb{A}_1=\{b_{11}^L,b_{12}^L\},\mathbb{A}_2=\{b_{13}^L,b_{21}^L,b_{22}^L\}$	0.011	47.51
$\mathbb{A}_1=\{b_{11}^L,b_{22}^L\},\mathbb{A}_2=\{b_{12}^L,b_{13}^L,b_{21}^L\}$	0.132	47.46
$\mathbb{A}_1=\{b_{13}^L,b_{22}^L\},\mathbb{A}_2=\{b_{11}^L,b_{12}^L,b_{21}^L\}$	0.165	47.39
$\mathbb{A}_1=\{b_{11}^L,b_{21}^L\},\mathbb{A}_2=\{b_{12}^L,b_{13}^L,b_{22}^L\}$	0.293	45.09
$\mathbb{A}_1=\{b_{11}^L,b_{13}^L\},\mathbb{A}_2=\{b_{12}^L,b_{21}^L,b_{22}^L\}$	0.294	45.89
$\mathbb{A}_1=\{b_{12}^L,b_{13}^L\},\mathbb{A}_2=\{b_{11}^L,b_{21}^L,b_{22}^L\}$	0.308	45.00
$\mathbb{A}_1=\{b_{11}^L\},\mathbb{A}_2=\{b_{12}^L,b_{13}^L,b_{21}^L,b_{22}^L\}$	0.470	41.80
$\mathbb{A}_1=\{b_{12}^L,b_{22}^L\},\mathbb{A}_2=\{b_{11}^L,b_{13}^L,b_{21}^L\}$	0.471	41.96
$\mathbb{A}_1=\{b_{13}^L,b_{21}^L\},\mathbb{A}_2=\{b_{11}^L,b_{12}^L,b_{22}^L\}$	0.590	39.97
$\mathbb{A}_1=\{b_{21}^L,b_{22}^L\},\mathbb{A}_2=\{b_{11}^L,b_{12}^L,b_{13}^L\}$	0.752	37.02
$\mathbb{A}_1=\{b_{13}^L\},\mathbb{A}_2=\{b_{11}^L,b_{12}^L,b_{21}^L,b_{22}^L\}$	0.767	36.83
$\mathbb{A}_1=\{b_{12}^L,b_{21}^L\},\mathbb{A}_2=\{b_{11}^L,b_{13}^L,b_{22}^L\}$	0.895	34.94
$\mathbb{A}_1=\{b_{22}^L\},\mathbb{A}_2=\{b_{11}^L,b_{12}^L,b_{13}^L,b_{21}^L\}$	0.929	34.30
$\mathbb{A}_1=\{b_{12}^L\},\mathbb{A}_2=\{b_{11}^L,b_{13}^L,b_{21}^L,b_{22}^L\}$	1.072	32.58
$\mathbb{A}_1=\{b_{21}^L\},\mathbb{A}_2=\{b_{11}^L,b_{12}^L,b_{13}^L,b_{22}^L\}$	1.353	29.48

The approximate monotonicity is useful to eliminate branches of infeasible partial assignments in the assignment process. From Table VI, we know that at stage 2, if line-side buffer b_{13}^L is assigned to the driver in charge of line-side buffer b_{11}^L (driver d_1), the estimated system throughput under partial assignment $A=(\mathbb{A}_1,\mathbb{A}_2)$, where $\mathbb{A}_1=\{b_{11}^L,b_{13}^L\}$, $\mathbb{A}_2=\emptyset$, is 45.99 (jph), which is less than TP_0. No matter which driver other line-side buffers are assigned to, partial assignments constructed based on this infeasible one are all infeasible because of the monotonicity of the throughput in the assignment process. So, at stage 2, the branch of assigning line-side buffer b_{13}^L to driver d_1 can be eliminated, which is shown in Fig. 7. Similarly, at stage 4, the branch of assigning line-side buffer b_{12}^L to driver d_2 can be eliminated. In Fig. 7, blocks in line with a number or X inside represent sorted line-side buffers in decreasing order of their workloads, where the number in the block denotes the index of the driver corresponding line-side buffer assigned to, and X indicates the line-side buffer has not been assigned to drivers yet. Solid arrows in the stages illustrate the process of assigning line-side buffers to drivers in Phase one of the SAB algorithm.

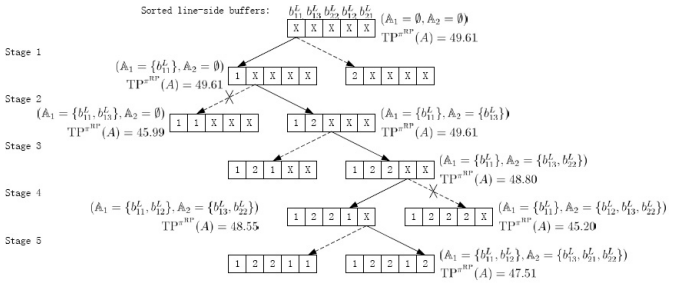


Fig. 7. The branching and branch elimination in Phase one

B. Case Study

The Sequential Assignment with Backtracking (SAB) algorithm is used to solve the LBAP problem in a real system of

general assembly line with material handling in automotive plant. The structure of the real system is the same as that of the system in [5], which is more complex than that in Fig. 1. In the system, more than 300 line-side buffers are assigned to 16 drivers by trial-and-error methods based on engineer's experience. The simulation model used to evaluate the system throughput is based on the method in [5]. $10,000$ minutes (about 21 subsequent shifts) including $5,000$ minutes warm-up time of the production is simulated and the simulation is run for 20 replications. Under the assignment in the plant, the estimated throughput is 29.81[1] jobs per hour (jph). In addition, the ideal throughput that all line-side buffers have sufficient part supply is $\mathrm{TP}_{\max} = 30.04$ (jph) and TP_0 is set as $0.9\mathrm{TP}_{\max} = 27.04$ (jph).

First, we calculate the lower bound of the lean number of drivers in the material handling subsystem. Workloads of line-side buffers are calculated in terms of (40) and total workload of line-side buffers is 24.720 (min), thus, the lower bound of the lean number of drivers is $\widehat{D}_{lean} = \left\lceil \frac{27.04*24.720}{60} \right\rceil = \lceil 11.14 \rceil = 12$. We set $\epsilon = 0.01$ and the maximum number of random transitions $n = 10,000$.

Then, the LBAP problem with $D = 12$ drivers is solved by the SAB algorithm. The algorithm is implemented by Visual Studio 2005 and runs on a PC with Pentium D 2.80GHz CPU, 2.00GB RAM, and Microsoft Windows XP system. The algorithm provides an assignment of feasibility probability $P_A = 0.952$ and visits it twice in the total 106 iterations in Phase two. Under this assignment, the system throughput is 27.08 (jph), which implies the algorithm can find a feasible assignment with lean number of drivers. The performance of the system under the assignment in factory and under the one the SAB algorithm finds is shown in Table VII. The SAB algorithm can also provide an observed feasible assignment for each $D = 13, 14, 15, 16$. If we set $\mathrm{TP}_0 = 0.95\mathrm{TP}_{\max} = 28.54$ (jph), we have $\widehat{D}_{lean} = \left\lceil \frac{28.54*24.720}{60} \right\rceil = \lceil 11.76 \rceil = 12$. In this case, the algorithm provides an observed feasible assignment for each $D = 13, 14, 15, 16$, while fails to find an observed feasible assignment for $D = 12$. The reason may be either $D = 12$ is not feasible or the maximum number of random transitions n is not large enough. The performance of the system under the assignments with $D = 13, 14, 15, 16$ is also shown in Table VII.

TABLE VII
COMPARISON OF ASSIGNMENTS

Assignment	D	$\mathrm{TP}^{\pi^{\mathrm{RP}}}(A)$[1]	P_A	n_A	#Itr[2]	#Time[3]
A^{SAB}	12*	27.08 ± 0.05	0.952	2	106	3.5
	13	28.59 ± 0.05	0.974	2	204	6.4
	14	29.73 ± 0.06	≈ 1	1	0	1.6
	15	29.95 ± 0.07	≈ 1	1	0	2.2
	16	30.00 ± 0.05	≈ 1	1	0	2.0
$A^{factory}$	16	29.81 ± 0.05	–	–	–	–

[1] Measured by jobs per hour, and the confidence level is 95%
[2] Total number of iterations in Phase two
[3] Total run time, measured by hour
* TP_0 is set as $0.9\mathrm{TP}_{\max} = 27.04$ (jph)

[1]To facilitate the presentation, in this subsection, throughputs are all biased by a scaling factor and total workload of lien-side buffers are biased by another scaling factor.

From Table VII, we can see the SAB algorithm finds a feasible assignment with 13 drivers even if $\mathrm{TP}_0 = 0.95\mathrm{TP}_{\max}$, which means, compared with the assignment implemented in the factory, 3 drivers can be saved, and meanwhile, the throughput keeps close to the ideal throughput (the gap is less than 5%). Under the assignment in factory, the difference among drivers' utilizations could be as large as 0.46, larger than half of the average utilization; while it is lower than 0.06 under the assignments the SAB algorithm provides. Because of the algorithm's capability of balancing drivers' utilizations, fewer drivers are needed to achieve the required throughput in the system of general assembly line with material handling.

The SAB algorithm is effective for solving the real LBAP problem. It provides feasible assignments for the real LBAP problem with high probability within several hours (almost all of the time is spent on performance evaluation of the system by simulation and the run time of the algorithm mainly depends on the number of drivers and the density of feasible assignments in the solution space). From Subsection IV-D, we know the SAB algorithm outperforms GA when the density of feasible assignment is $\frac{1}{15}$. It is worth noting that the density of feasible assignments in the real LBAP problem is much rare (it is estimated to be lower than 10^{-6} for $D = 16$ and $\mathrm{TP}_0 = 0.95\mathrm{TP}_{\max}$) and it would be much lower than 10^{-6} for $D = 13$ because it is close to the lean number of drivers. It is because the SAB algorithm takes advantages of the structural characteristics developed in Section V, it is effective for solving the LBAP problem.

In addition, the SAB algorithm is suitable to assign a subset of line-side buffers to drivers, e.g., some line-side buffers must be reassigned in the case of driver's absenteeism. In this situation, only line-side buffers in the charge of the absentees are to be assigned to other drivers. The assignment process is the same as that of assigning all line-side buffers to all drivers.

VII. CONCLUSIONS

The Line-side Buffer Assignment Problem (LBAP) in the system of general assembly line with material handling is a difficult problem. In this paper, the fixed zoning version of the LBAP problem is formulated, proved to be NP-hard, and solved by the Sequential Assignment with Backtracking (SAB) algorithm, which is developed based on two structural characteristics of the problem–the analogousness between the LBAP problem and the Parallel Machine Scheduling (PMS) problem and the monotonicity property of the throughput in the course of assigning line-side buffers to drivers. The global convergence of the SAB algorithm is established when feasible assignments exist for the problem. The algorithm is tested on a real system, and the results demonstrate that it is effective to the LBAP problem. Although the focus of the paper is on the general assembly line, the formulation, the complexity result, and the solution methodology can be easily extended to systems containing disassembly and/or rework operations since our method is based on simulation model and these systems share similar features with the systems of general assembly line. The study of flexible zoning line-side buffer assignment problem is also possible future work.

APPENDIX A
NOTATIONS FOR THE SYSTEM STATE

$\alpha^m_{u,t}$ $\alpha^m_{u,t} = 1$ if machine m_u, at time t, is operational (viz., up and busy); $\alpha^m_{u,t} = -1$ if machine m_u is idle (viz., up but forced down); $\alpha^m_{u,t} = 0$ if machine m_u is down, $u = 1, 2, \ldots, M$.

α^m_t $\alpha^m_t = (\alpha^m_{1,t}, \alpha^m_{2,t}, \ldots, \alpha^m_{M,t})$.

$T^{m,opr}_{u,t}$ Accumulated operation time from the beginning of latest resumption time of machine m_u for operation to time t if $\alpha^m_{u,t} = 1$, $u = 1, 2, \ldots, M$.

$T^{m,opr}_t$ $T^{m,opr}_t = (T^{m,opr}_{1,t}, T^{m,opr}_{2,t}, \ldots, T^{m,opr}_{M,t})$.

$T^{m,idle}_{u,t}$ Accumulated idle time from the beginning of latest forced down of machine m_u to time t if $\alpha^m_{u,t} = -1$, $u = 1, 2, \ldots, M$.

$T^{m,idle}_t$ $T^{m,idle}_t = (T^{m,idle}_{1,t}, T^{m,idle}_{2,t}, \ldots, T^{m,idle}_{M,t})$.

$T^{m,up}_{u,t}$ Accumulated uptime from the end of latest repair of machine m_u to time t if $\left|\alpha^m_{u,t}\right| = 1$, $u = 1, 2, \ldots, M$.

$T^{m,up}_t$ $T^{m,up}_t = (T^{m,up}_{1,t}, T^{m,up}_{2,t}, \ldots, T^{m,up}_{M,t})$.

$T^{m,down}_{u,t}$ Accumulated downtime from the beginning of latest repair of machine m_u to time t if $\alpha^m_{u,t} = 0$, $u = 1, 2, \ldots, M$.

$T^{m,down}_t$ $T^{m,down}_t = (T^{m,down}_{1,t}, T^{m,down}_{2,t}, \ldots, T^{m,down}_{M,t})$.

s^{Ib}_t State vector of in-process buffers (excluding the input and output in-process buffers) at time t. It is a $M-1$-by-1 vector whose element $s^{Ib}_{u,t}$ is the inventory level of in-process buffer b^I_u at time t. $s^{Ib}_{u,t}$ is an integer and $0 \leqslant s^{Ib}_{u,t} \leqslant C^I_u$, $u = 1, 2, \ldots, M-1$.

$L^{Lb}_{uv,t}$ Inventory level of line-side buffer $b^L_{uv} \in \mathbb{B}$ at time t. $L^{Lb}_{uv,t}$ is an integer and $0 \leqslant L^{Lb}_{uv,t} \leqslant C^L_{uv}$.

L^{Lb}_t $L^{Lb}_t = (L^{Lb}_{uv,t})_{b^L_{uv} \in \mathbb{B}}$.

$T^{Lb}_{uv,t}$ Accumulated time from the beginning of current duration of parts shortage in line-side $b^L_{uv} \in \mathbb{B}$ to time t if the line-side buffer is still short of parts at time t.

T^{Lb}_t $T^{Lb}_t = (T^{Lb}_{uv,t})_{b^L_{uv} \in \mathbb{B}}$.

$\alpha^d_{i,t}$ $\alpha^d_{i,t} = b^L_{uv}$ If driver d_i, at time t, is delivering parts for line-side buffer $b^L_{uv} \in \mathbb{A}_i$; $\alpha^d_{i,t} = 0$ if driver d_i is idle, $i = 1, 2, \ldots, D$.

α^d_t $\alpha^d_t = (\alpha^d_{1,t}, \alpha^d_{2,t}, \ldots, \alpha^d_{D,t})$.

$T^{d,busy}_{i,t}$ Accumulated busy time of driver d_i from the beginning of current trip to time t if $\alpha^d_{i,t} \neq 0$, $i = 1, 2, \ldots, D$.

$T^{d,busy}_t$ $T^{d,busy}_t = (T^{d,busy}_{1,t}, T^{d,busy}_{2,t}, \ldots, T^{d,busy}_{D,t})$.

$T^{d,idle}_{i,t}$ Accumulated idle time of driver d_i from the end of last trip to time t if $\alpha^d_{i,t} = 0$, $i = 1, 2, \ldots, D$.

$T^{d,idle}_t$ $T^{d,idle}_t = (T^{d,idle}_{1,t}, T^{d,idle}_{2,t}, \ldots, T^{d,idle}_{D,t})$.

s^m_t State vector of machines at time t. $s^m_t = (\alpha^m_t, T^{m,opr}_t, T^{m,idle}_t, T^{m,up}_t, T^{m,down}_t)$.

s^{Lb}_t State vector of line-side buffers at time t. $s^{Lb}_t = (L^{Lb}_t, T^{Lb}_t)$.

s^d_t State vector of drivers at time t. $s^d_t = (\alpha^d_t, T^{d,busy}_t, T^{d,idle}_t)$.

APPENDIX B
PROOFS OF PROPOSITIONS IN SECTION V

In this appendix, Propositions 5.1, 5.2, 5.3, and 5.5, are proved. It is worth noting that these two propositions are proved based on assumption of Blocking-After-Service (BAS) mechanism. It is similar to draw the conclusions for the blocking mechanism of Blocking-Before-Service (BBS).

A. Proof of Proposition 5.1

Proof: In Subsection II-D1, the dynamics of the assembly subsystem is developed. Now we will prove Proposition 5.1 by justification of arrival times of all jobs at b^I_M being not delayed, which can be proved by induction based on the recursive equations presented in Subsection II-D. Recall that the general figure of a segment of the assembly subsystem is shown in Fig. 3.

1) We will prove the exit time of the 1^{st} job from in-process buffer $b^I_{u_j}$, $j = 1, 2, \ldots, N^I_u$, and the arrival time at in-process buffer b^I_u, $u = 1, 2, \ldots, M$, are not delayed. This will be proved also by induction.
   a) First, we prove exit times of the 1^{st} job from input in-process buffers b^I_0's and arrival times at b^I_w's, which are the downstream in-process buffers of b^I_0's, are not delayed. For $k = 1$ and $u = w$, the related key timings are as follows:

$$T^e_0(1) = \max(T^a_0(1), T^a_w(0)) = 0$$

$$T^r_{wv}(1) = \max_{0<q\leqslant Q^C_{1,wv}} T^{pr}_{wv}(q), b^L_{wv} \in \mathbb{B}$$

$$T^s_w(1) = \max(T^e_0(1), \max_{1\leqslant v\leqslant N^L_w} T^r_{wv}(1)) = \max_{1\leqslant v\leqslant N^L_w} T^r_{wv}(1)$$

$$T^f_w(1) = T^s_w(1) + T^p_w(1)$$

$$T^a_w(1) = \max(T^f_w(1), T^e_w(1 - C^I_w)) = T^f_w(1)$$

   It is clear that $\widetilde{T}^e_0(1) = T^e_0(1) = 0$. In addition, since $\widetilde{T}^{pr}_{wv}(q) \leqslant T^{pr}_{wv}(q)$, where $0 < q \leqslant Q^C_{1,wv}$, $v = 1, 2, \ldots, N^L_w$, we have $\widetilde{T}^r_{wv}(1) \leqslant T^r_{wv}(1)$, $\widetilde{T}^s_w(1) \leqslant T^s_w(1)$, $\widetilde{T}^f_w(1) \leqslant T^f_w(1)$ and $\widetilde{T}^a_w(1) \leqslant T^a_w(1)$. Thus, exit times of the 1^{st} job from b^I_0's and arrival times at b^I_w's are not delayed.
   b) The arrival time of the 1^{st} job at in-process buffer $b^I_{u_j}$, $j = 1, 2, \ldots, N^I_u$, $u = 1, 2, \ldots, M$, is assumed to be not delayed, i.e., $\widetilde{T}^a_{u_j}(1) \leqslant T^a_{u_j}(1)$. In terms of

$$T^e_{u_j}(1) = \max(T^a_{u_j}(1), T^a_u(0)) = T^a_{u_j}(1), \quad j = 1, 2, \ldots, N^I_u$$

$$T^r_{uv}(1) = \max_{0<q\leqslant Q^C_{1,uv}} T^{pr}_{uv}(q), b^L_{wv} \in \mathbb{B}$$

$$T^s_u(1) = \max(\max_{1\leqslant j\leqslant N^I_u} T^e_{u_j}(1), \max_{1\leqslant v\leqslant N^L_u} T^r_{uv}(1))$$

$$T^f_u(1) = T^s_u(1) + T^p_u(1)$$

$$T^a_u(1) = \max(T^f_u(1), T^e_u(1 - C^I_u))$$

   we have $\widetilde{T}^e_{u_j}(1) \leqslant T^e_{u_j}(1)$. And due to $\widetilde{T}^{pr}_{uv}(q) \leqslant T^{pr}_{uv}(q)$, where $0 < q \leqslant Q^C_{1,uv}$, $v = 1, 2, \ldots, N^L_u$,

we have $\widetilde{T}^r_{uv}(1) \leqslant T^r_{uv}(1)$, $\widetilde{T}^s_u(1) \leqslant T^s_u(1)$, $\widetilde{T}^f_u(1) \leqslant T^f_u(1)$ and $\widetilde{T}^a_u(1) \leqslant T^a_u(1)$. That is to say, the exit time of the 1^{st} job from in-process buffer $b^I_{u_j}$, $j = 1, 2, \ldots, N^I_u$, and the arrival time at in-process buffer b^I_u, are not delayed.

c) From 1a) and 1b), we can draw the conclusion that $\widetilde{T}^e_{u-1}(1) \leqslant T^e_{u-1}(1)$ and $\widetilde{T}^a_u(1) \leqslant T^a_u(1)$, $u = 1, 2, \ldots, M$.

2) Exit times of the p^{th} job from in-process buffer b^I_{u-1}'s and arrival times at in-process buffer b^I_u's, $p = 1, 2, \ldots, k-1$, $u = 1, 2, \ldots, M$, are assumed to be not delayed, i.e., $\widetilde{T}^e_{u-1}(p) \leqslant T^e_{u-1}(p)$ and $\widetilde{T}^a_u(p) \leqslant T^a_u(p)$, then

a) First, we will prove exit times of the k^{th} job from in-process buffer b^I_0's and arrival times at b^I_w's, which are the downstream in-process buffers of b^I_0's, are not delayed. The related key timings are:

$$\begin{aligned} T^e_0(k) &= \max(T^a_0(k), T^a_w(k-1)) \\ &= T^a_w(k-1),\ k = 1, 2, \ldots \end{aligned}$$

$$\begin{aligned} T^r_{wv}(k) = &\max_{\sum_{j=1}^{k-1} Q^C_{j,wv} < q \leqslant \sum_{j=1}^{k} Q^C_{j,wv}} T^{pr}_{wv}(q), \\ &b^L_{wv} \in \mathbb{B},\ k = 1, 2, \ldots \end{aligned}$$

$$\begin{aligned} T^s_w(k) = \max(&\max_{1 \leqslant j \leqslant N^I_w} T^e_{w_j}(k), \\ &\max_{1 \leqslant v \leqslant N^L_w} T^r_{wv}(k)),\ k = 1, 2, \ldots \end{aligned}$$

$$T^f_w(k) = T^s_w(k) + T^p_w(k),\ k = 1, 2, \ldots$$

$$\begin{aligned} T^a_w(k) &= \max(T^f_w(k), T^e_w(k - C^I_w)), \\ &k = 1, 2, \ldots \end{aligned}$$

Since $\widetilde{T}^a_w(k-1) \leqslant T^a_w(k-1)$, we have $\widetilde{T}^e_0(k) \leqslant T^e_0(k)$. And due to $\widetilde{T}^{pr}_{wv}(q) \leqslant T^{pr}_{wv}(q)$ and $\widetilde{T}^e_w(k - C^I_w) \leqslant T^e_w(k - C^I_w)$, where $\sum_{j=1}^{k-1} Q^C_{j,wv} < q \leqslant \sum_{j=1}^{k} Q^C_{j,wv}$, $v = 1, 2, \ldots, N^L_w$, we have $\widetilde{T}^s_w(k) \leqslant T^s_w(k)$, $\widetilde{T}^f_w(k) \leqslant T^f_w(k)$ and $\widetilde{T}^a_w(k) \leqslant T^a_w(k)$. Thus, exit times of the k^{th} job from b^I_0's and arrival times at b^I_w's are not delayed.

b) The arrival time of the k^{th} job at in-process buffer $b^I_{u_j}$, $j = 1, 2, \ldots, N^I_u$, $u = 1, 2, \ldots, M$, is assumed to be not delayed, i.e., $\widetilde{T}^a_{u_j}(k) \leqslant T^a_{u_j}(k)$. In terms of

$$\begin{aligned} T^e_{u_j}(k) &= \max(T^a_{u_j}(k), T^a_u(k-1)), \\ &j = 1, 2, \ldots, N^I_u,\ k = 1, 2, \ldots \end{aligned}$$

$$\begin{aligned} T^r_{uv}(k) = &\max_{\sum_{j=1}^{k-1} Q^C_{j,uv} < q \leqslant \sum_{j=1}^{k} Q^C_{j,uv}} T^{pr}_{uv}(q), \\ &b^L_{uv} \in \mathbb{B},\ k = 1, 2, \ldots \end{aligned}$$

$$\begin{aligned} T^s_u(k) = \max(&\max_{1 \leqslant j \leqslant N^I_w} T^e_{u_j}(k), \\ &\max_{1 \leqslant v \leqslant N^L_w} T^r_{uv}(k)),\ k = 1, 2, \ldots \end{aligned}$$

$$T^f_u(k) = T^s_u(k) + T^p_u(k),\ k = 1, 2, \ldots$$

$$\begin{aligned} T^a_u(k) &= \max(T^f_u(k), T^e_u(k - C^I_u)) \\ &k = 1, 2, \ldots \end{aligned}$$

we have $\widetilde{T}^e_{u_j}(k) \leqslant T^e_{u_j}(k)$, $\widetilde{T}^s_u(k) \leqslant T^s_u(k)$, $\widetilde{T}^f_u(k) \leqslant T^f_u(k)$ and $\widetilde{T}^a_u(k) \leqslant T^a_u(k)$.

c) From 2a) and 2b), we have $\widetilde{T}^e_{u-1}(k) \leqslant T^e_{u-1}(k)$ and $\widetilde{T}^a_u(k) \leqslant T^a_u(k)$, $u = 1, 2, \ldots, M$, $k = 1, 2, \ldots$

3) From 1) and 2), we draw the conclusion that exit times of all finished products produced in time interval $[0, T]$ from buffers b^I_{u-1}'s and arrival times at b^I_u's, $u = 1, 2, \ldots, M$, are not delayed.

Thus the conclusion holds. ■

B. Proof of Proposition 5.2

Proof: In Subsection II-C, we know if decision rule π_t determines driver d_i to serve line-side buffer $b^L_{uv} \in \mathbb{A}_i$ in his/her l^{th} trip, then $a_{i,t} = (\pi_t(X_t))_i = b^L_{uv}$ and $T^{dp}_{i,a_{i,t}}(l) = t$; in other words, $T^{dp}_{i,a_{i,t}}(l)$ is determined by the delivery policy π, $i = 1, 2, \ldots, D$, $l = 1, 2, \ldots$, $t \geqslant 0$. In the material handling subsystem, driver d_{i_0}'s departure times generated by the feasible delivery policy π is shown in Fig. 8. In the following, we construct a feasible policy $\widetilde{\pi}$, under which $\forall b^L_{uv} \in \mathbb{B}$, $\widetilde{T}^{pr}_{uv}(q) \leqslant T^{pr}_{uv}(q)$, $q = 1, 2, \ldots$

The delivery policy $\widetilde{\pi}$ is such a policy that $\widetilde{\pi}(\widetilde{X}_t) = \pi(X_t)$ except $(\widetilde{\pi}_t(\widetilde{X}_t))_{i_0} = a^*$ when $(\pi_t(X_t))_{i_0} = b^L_{u_0v_0}$. Driver d_{i_0}'s departure times generated by $\widetilde{\pi}$ is also shown in Fig. 8, in which the trip with dashed line is virtual.

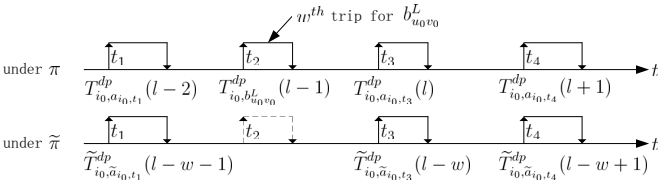


Fig. 8. The process of driver d_{i_0}'s delivering parts for line-side buffers

Obviously, for line-side buffer $b^L_{u_0v_0}$, $\widetilde{T}^{pr}_{u_0v_0}(q) = 0 \leqslant T^{pr}_{u_0v_0}(q)$ for all q. Thus, in the following, we focus on line-side buffer $b^L_{uv} \in \mathbb{B} \setminus \{b^L_{u_0v_0}\}$.

1) At the beginning of the production, all line-side buffers are empty of parts. We assume, under delivery policy π, driver d_{i_0} has served line-side buffer $b^L_{u_0v_0}$ for w trips before his/her first trip for a certain line-side buffer, say, $b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$. From the definition of $\widetilde{\pi}$, we have $\widetilde{T}^{dp}_{i_0,b^L_{uv}}(l-w) = T^{dp}_{i_0,b^L_{uv}}(l)$. In terms of

$$\begin{aligned} T^{pd}_{i_0,b^L_{uv}}(l) &= T^{dp}_{i_0,b^L_{uv}}(l) + \frac{T^T_{l,uv}}{2},\ b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0} \\ T^{pr}_{uv}(q) &= T^{pd}_{i_0,b^L_{uv}}(l),\ b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}, \\ &q = 1, 2, \ldots, Q^D_{l,uv} \end{aligned}$$

under delivery policy π and

$$\begin{aligned} \widetilde{T}^{pd}_{i_0,b^L_{uv}}(l-w) &= \widetilde{T}^{dp}_{i_0,b^L_{uv}}(l-w) + \frac{\widetilde{T}^T_{l-w,uv}}{2}, \\ &b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0} \end{aligned}$$

$$\widetilde{T}^{pr}_{uv}(q) = \widetilde{T}^{pd}_{i_0,b^L_{uv}}(l-w),\ b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0},$$
$$q = 1, 2, \ldots, \widetilde{Q}^D_{l-w,uv}$$

under $\widetilde{\pi}$, and noting that $\widetilde{T}^T_{l-w,uv} = T^T_{l,uv}$, we have $\widetilde{T}^{pr}_{uv}(q) \leqslant T^{pr}_{uv}(q)$, $q = 1, 2, \ldots, Q^D_{l,uv}$. It means release times of parts delivered in the first delivery for line-side buffer $b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$ are not delayed. It is similar to prove release times of parts delivered in the first trip for line-side buffer $b^L_{uv} \in \mathbb{B} \setminus \mathbb{A}_{i_0}$ are not delayed.

2) In Proof of Proposition (5.1), we know $\widetilde{T}^e_u(k) \leqslant T^e_u(k)$ if release times of delivered parts in all line-side buffers are not delayed, $u = 0, 1, \ldots, M-1$. For line-side buffer $b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$, we have

$$T^r_{uv}(k) = \max_{\sum_{j=1}^{k-1} Q^C_{j,uv} < q \leqslant \sum_{j=1}^{k} Q^C_{j,uv}} T^{pr}_{uv}(q)$$
$$T^s_u(k) = \max(\max_{1 \leqslant j \leqslant N^I_u} T^e_{u_j}(k), \max_{1 \leqslant v \leqslant N^L_u} T^r_{uv}(k))$$
$$T^{pc}_{uv}(q) = T^s_u(k),\ q = \sum_{j=1}^{k-1} Q^C_{j,uv} + 1,$$
$$\sum_{j=1}^{k-1} Q^C_{j,uv} + 2, \ldots, \sum_{j=1}^{k} Q^C_{j,uv}$$

under delivery policy π and similarly,

$$\widetilde{T}^r_{uv}(k) = \max_{\sum_{j=1}^{k-1} Q^C_{j,uv} < q \leqslant \sum_{j=1}^{k} Q^C_{j,uv}} \widetilde{T}^{pr}_{uv}(q)$$
$$\widetilde{T}^s_u(k) = \max(\max_{1 \leqslant j \leqslant N^I_u} \widetilde{T}^e_{u_j}(k), \max_{1 \leqslant v \leqslant N^L_u} \widetilde{T}^r_{uv}(k))$$
$$\widetilde{T}^{pc}_{uv}(q) = \widetilde{T}^s_u(k),\ q = \sum_{j=1}^{k-1} Q^C_{j,uv} + 1,$$
$$\sum_{j=1}^{k-1} Q^C_{j,uv} + 2, \ldots, \sum_{j=1}^{k} Q^C_{j,uv}$$

under $\widetilde{\pi}$. It is easy to see $\widetilde{T}^{pc}_{uv}(q) \leqslant T^{pc}_{uv}(q)$ holds, $q = 1, 2, \ldots, \sum_{j=1}^{k} Q^C_{j,uv}$, where k satisfies $\sum_{j=1}^{k} Q^C_{j,uv} \leqslant Q^D_{l,uv} < \sum_{j=1}^{k+1} Q^C_{j,uv}$. It implies the times that delivered parts are consumed are not delayed. In the following, we will prove release times of parts to be delivered for line-side buffer $b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$ are not delayed. Clearly, we have

$$T^{pd}_{i_0,b^L_{uv}}(l) = \max(T^{dp}_{i_0,b^L_{uv}}(l) + \frac{T^T_{l,uv}}{2},$$
$$T^{pc}_{uv}(Q^D_{l,uv} - C^L_{uv})),\ b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$$
$$T^{pr}_{uv}(q) = T^{pd}_{i_0,b^L_{uv}}(l),\ q = Q^D_{l-1,uv} + 1,$$
$$Q^D_{l-1,uv} + 2, \ldots, Q^D_{l,uv},\ b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$$

under delivery policy π and similarly,

$$\widetilde{T}^{pd}_{i_0,b^L_{uv}}(l-w) = \max(\widetilde{T}^{dp}_{i_0,b^L_{uv}}(l-w) + \frac{\widetilde{T}^T_{l-w,uv}}{2},$$
$$\widetilde{T}^{pc}_{uv}(\widetilde{Q}^D_{l-w,uv} - C^L_{uv})),$$
$$b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$$
$$\widetilde{T}^{pr}_{uv}(q) = \widetilde{T}^{pd}_{i_0,b^L_{uv}}(l-w),\ q = \widetilde{Q}^D_{l-w-1,uv} + 1,$$
$$\widetilde{Q}^D_{l-w-1,uv} + 2, \ldots, \widetilde{Q}^D_{l-w,uv},$$
$$b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$$

under $\widetilde{\pi}$. It is clear that $\widetilde{Q}^D_{l-w,uv} = Q^D_{l,uv}$. Since $\widetilde{T}^{pc}_{uv}(\widetilde{Q}^D_{l-w,uv} - C^L_{uv}) \leqslant T^{pc}_{uv}(Q^D_{l,uv} - C^L_{uv})$ and $\widetilde{T}^{dp}_{i_0,b^L_{uv}}(l-w) = T^{dp}_{i_0,b^L_{uv}}(l)$, and noting that $\widetilde{T}^T_{l-w,uv} = T^T_{l,uv}$, we have $\widetilde{T}^{pd}_{i_0,b^L_{uv}}(l-w) \leqslant T^{pd}_{i_0,b^L_{uv}}(l)$. Thus, $\widetilde{T}^{pr}_{uv}(q) \leqslant T^{pr}_{uv}(q)$, $q = Q^D_{l-1,uv} + 1, Q^D_{l-1,uv} + 2, \ldots, Q^D_{l,uv}$. It means release times of parts to be delivered in line-side buffer $b^L_{uv} \in \widetilde{\mathbb{A}}_{i_0}$ are not delayed. It is similar to prove release times of parts to be delivered in line-side buffer $b^L_{uv} \in \mathbb{B} \setminus \mathbb{A}_{i_0}$ are not delayed.

3) In terms of 1) and 2), $\widetilde{T}^{pr}_{uv}(q) \leqslant T^{pr}_{uv}(q)$ for all q, $b^L_{uv} \in \mathbb{B} \setminus \{b^L_{u_0v_0}\}$.

Thus, under delivery policy $\widetilde{\pi}$, release times of all parts are not delayed. And clearly, delivery policy $\widetilde{\pi}$ is feasible. Therefore, the conclusion holds. ■

C. Proof of Proposition 5.3

Proof: In the k^{th} iteration of the second phase, $A(k+1)$ could only be the child partial assignment $A_i(k)$ and the parent partial assignment $A'(k)$ of the current partial assignment $A(k)$ and depends on the current samples of the system throughput under $A_i(k)$. Since $A_i(k)$ and $A'(k)$ in turn depend on $A(k)$, it is clear that $A(k+1)$ only depends on $A(k)$. Thus $\{A(k)\}_{k=1}^{\infty}$ is a Markov chain. Obviously, the state space of the Markov chain is $\mathscr{A}$, which is the set of all possible partial assignments that could be visited by the SAB algorithm. It is also clear that $\{A(k)\}_{k=1}^{\infty}$ is a finite-state Markov chain.

As we stated in Remark 5.2, for any partial assignment $A \in \mathscr{A}$, there exists some $\kappa \leqslant N$ such that the κ-step transition probabilities $P^{(\kappa)}(A, A_0) > 0$ and $P^{(\kappa)}(A_0, A) > 0$. That is to say, for any partial assignments $A, \widetilde{A} \in \mathscr{A}$, there exist some $\kappa_1, \kappa_2 \leqslant N$ such that $P^{(\kappa_1)}(A, A_0) > 0$ and $P^{(\kappa_2)}(A_0, \widetilde{A}) > 0$. Thus, $P^{(\kappa_1+\kappa_2)}(A, \widetilde{A}) > 0$. In other words, all states of the Markov chain are in the same communicating class and the Markov chain is irreducible. Since it is a finite-state Markov chain, all states are positive recurrent. In addition, for $A \in \mathscr{A}_0$, $P(A, A) > 0$, which implies the Markov chain is aperiodic. Thus, the Markov chain is ergodic and hence has a unique stationary distribution $\{\nu(A)\}_{A \in \mathscr{A}}$.

Furthermore, Markov chain $\{A(k)\}_{k=1}^{\infty}$ is time reversible. The proof is as follows.

For the partial assignment $\widehat{A}$ whose child partial assignments are assignments $\widehat{A}_i \in \mathscr{A}_0$, $i = 1, 2, \ldots, D$, we have

$$\nu(\widehat{A}_i) = \nu(\widehat{A})P(\widehat{A}, \widehat{A}_i) + \nu(\widehat{A}_i)P(\widehat{A}_i, \widehat{A}_i) \tag{59}$$

because $\{\nu(A)\}_{A \in \mathscr{A}}$ is the stationary distribution of Markov chain $\{A(k)\}_{k=1}^{\infty}$. In other words, we have

$$\nu(\widehat{A}_i)P(\widehat{A}_i, \widehat{A}) = \nu(\widehat{A})P(\widehat{A}, \widehat{A}_i), i = 1, 2, \ldots, D, \tag{60}$$

which is the detailed balance condition.

In addition, the following equation holds:

$$\nu(\widehat{A}) = \nu(\widehat{A}')P(\widehat{A}', A) + \sum_{i=1}^{D} \nu(\widehat{A}_i)P(\widehat{A}_i, \widehat{A}), \tag{61}$$

where $\widehat{A}'$ is the parent partial assignment of $\widehat{A}$. From (60) and considering $P(\widehat{A},\widehat{A}')+\sum_{i=1}^{D}P(\widehat{A},\widehat{A}_i)=1$, we derive

$$\nu(\widehat{A})P(\widehat{A},\widehat{A}')=\nu(\widehat{A}')P(\widehat{A}',\widehat{A}), \tag{62}$$

which is also the detailed balance condition.

For other partial assignment $A\in\mathscr{A}$, it is easy to derive the above equation by induction. Thus, we have

$$\nu(A)P(A,A')=\nu(A')P(A',A),\forall A\in\mathscr{A}, \tag{63}$$

which implies

$$\nu(A)P(A,\widetilde{A})=\nu(\widetilde{A})P(\widetilde{A},A),\forall A,\widetilde{A}\in\mathscr{A} \tag{64}$$

because the one-step transition probability from a partial assignment to neither its parent or its children is 0. Thus, based on the Kolmogorov criterion [25], Markov chain $\{A(k)\}_{k=1}^{\infty}$ is time reversible. ■

D. Proof of Proposition 5.5

Proof: Let $\bar{A}$ be the closest common ancestor partial assignment of infeasible assignment $\widetilde{A}$ and feasible assignment A. Clearly, $\bar{A}$ is on the unique shortest path from $\widetilde{A}$ to A and that from A to $\widetilde{A}$, and $\kappa(\widetilde{A},\bar{A})=\kappa(\bar{A},\widetilde{A})=\kappa(\bar{A},A)=\kappa(A,\bar{A})$. In addition, in terms of Theorem 5.1, $\bar{A}$ is feasible because A is feasible.

From the transition probabilities defined in (54) and (55), we have

$$\begin{aligned}P(\widetilde{A},\widetilde{A}')&\geqslant 1-(\epsilon+\beta-2\epsilon\beta)\beta,\\ P^{\kappa(\widetilde{A}',\bar{A})}(\widetilde{A}',\bar{A})&\geqslant\epsilon^{\kappa(\widetilde{A}',\bar{A})},\\ P^{\kappa(\bar{A},A)}(\bar{A},A)&\geqslant(\frac{(1-\alpha-\epsilon+2\epsilon\alpha)(1-\alpha)}{D})^{\kappa(\bar{A},A)},\end{aligned} \tag{65}$$

and

$$\begin{aligned}P(A,A')&\leqslant 1-(1-\alpha-\epsilon+2\epsilon\alpha)(1-\alpha),\\ P^{\kappa(A',\bar{A})}(A',\bar{A})&\leqslant\{1-\frac{1}{D}(1-\alpha-\epsilon+2\epsilon\alpha)\\ &\qquad\cdot[1-\alpha+(D-1)\beta]\}^{\kappa(A',\bar{A})},\\ P^{\kappa(\bar{A},\widetilde{A})}(\bar{A},\widetilde{A})&\leqslant\beta(\frac{1-\epsilon}{D})^{\kappa(A',\bar{A})},\end{aligned} \tag{66}$$

where $\widetilde{A}'$ and A' are parent partial assignments of $\widetilde{A}$ and A respectively.

Based on $\alpha\leqslant\epsilon$ and $1-\beta(1-\alpha)(1-\epsilon)\leqslant(1-\alpha)^2$ in Assumption 5.3, we have $(1-\alpha)(1-\alpha-\epsilon+2\epsilon\alpha)\geqslant(1-\epsilon)[1-\beta(1-\alpha-\epsilon+2\epsilon\alpha)]$. Together with $\beta\leqslant\epsilon^{N-1}[1-\beta(\epsilon+\beta-2\epsilon\beta)]\leqslant\epsilon^{\kappa(\widetilde{A}',\bar{A})}[1-\beta(\epsilon+\beta-2\epsilon\beta)]$ and $1-\beta(1-\alpha-\epsilon+2\epsilon\alpha)\geqslant 1-(1-\alpha)(1-\alpha-\epsilon+2\epsilon\alpha)$, we have $P^{(\kappa(\widetilde{A},A))}(\widetilde{A},A)=P(\widetilde{A},\widetilde{A}')\cdot P^{\kappa(\widetilde{A}',\bar{A})}(\widetilde{A}',\bar{A})\cdot P^{\kappa(\bar{A},A)}(\bar{A},A)\geqslant P(A,A')\cdot P^{\kappa(A',\bar{A})}(A',\bar{A})\cdot P^{\kappa(\bar{A},\widetilde{A})}(\bar{A},\widetilde{A})=P^{(\kappa(A,\widetilde{A}))}(A,\widetilde{A})$, which completes the proof. ■

ACKNOWLEDGEMENT

We appreciate the helpful discussions from Prof. Xiaolan Xie, Prof. Leyuan Shi, and Mr. Yanjia Zhao, and careful review and detailed comments and suggestions from three anonymous reviewers for improving the manuscript.

REFERENCES

[1] A. Asef-Vaziri and G. Laporte, "Loop based facility planning and material handling," *European Journal of Operational Research*, vol. 164, no. 1, pp. 1–11, 2005.

[2] J. A. Buzacott and J. G. Shanthikumar, *Stochastic models of Manufacturing Systems*. Englewood Cliffs, N.J.: Prentice Hall, 1993.

[3] G. M. Buxey and D. Sadjadi, "Simulation studies of conveyor-paced assembly lines with buffer capacity," *International Journal of Production Research*, vol. 14, no. 5, pp. 607–624, 1976.

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

[5] 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*, CFINS, 2009. [Online]. Available: http://www.cfins.au.tsinghua.edu.cn/files/paper/TASE309_full_20091003

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

[7] J. Li and S. M. Meerkov, *Production Systems Engineering*. New York: Springer, 2009.

[8] J. Li, "Overlapping decomposition: A system-theoretic method for modeling and analysis of complex manufacturing systems," *IEEE Transactions on Automation Science and Engineering*, vol. 2, no. 1, pp. 40–53, 2005.

[9] S. B. Gershwin and J. E. Schor, "Efficient algorithms for buffer space allocation," *Annals of Operations Research*, vol. 93, no. 1-4, pp. 117–144, 2000.

[10] L. Shi and S. Men, "Optimal buffer allocation in production lines," *IIE Transactions*, vol. 35, no. 1, pp. 1–10, 2003.

[11] C. Becker and A. Scholl, "A survey on problems and methods in generalized assembly line balancing," *European Journal of Operational Research*, vol. 168, no. 3, pp. 694–715, 2006.

[12] N. Boysen, M. Fliedner, and A. Scholl, "A classification of assembly line balancing problems," *European Journal of Operational Research*, vol. 183, no. 2, pp. 674–693, 2007.

[13] R. Burkard, M. Dell'Amico, and S. Martello, *Assignment Problems*. Philadelphia: SIAM, 2009.

[14] D. W. Pentico, "Assignment problems: A golden anniversary survey," *European Journal of Operational Research*, vol. 176, no. 2, pp. 774–793, 2007.

[15] D. S. Johnson, A. Demers, J. D. Ullman, M. R. Garey, and R. L. Grahanm, "Worst-case performance bounds for simple one-dimensional packing algorithms," *SIAM Journal on Computing*, vol. 3, no. 4, pp. 299–325, 1974.

[16] E. Mokotoff, "Parallel machine scheduling problems: A survey," *Asia-Pacific Journal of Operational Research*, vol. 18, no. 2, pp. 193–242, 2001.

[17] J. E. G. Coffman, M. R. Garey, and D. S. Johnson, "An application of bin-packing to multiprocessor scheduling," *SIAM Journal on Computing*, vol. 7, no. 1, pp. 1–17, 1978.

[18] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: W.H. Freeman and Co., 1979.

[19] L. Shi and S. Ólafsson, "Nested partitions method for stochastic optimization," *Methodology and Computing in Applied Probability*, vol. 2, no. 3, pp. 271–291, 2000.

[20] ——, *Nested Partitions Method, Theorey and Applications*. New York: Springer, 2009.

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

[22] R. L. Graham, "Bounds for certain multiprocessing anomalies," *The Bell System Technical Journal*, vol. 45, no. 9, pp. 1563–1581, 1966.

[23] ——, "Bounds on multiprocessing timing anomalies," *SIAM Journal on Applied Mathematics*, vol. 17, no. 2, pp. 416–429, 1969.

[24] J. E. G. Coffman and G. S. Lueker, *Probabilistic Analysis of Packing and Partitioning Algorithms*. New York: Wiley, 1991.

[25] R. W. Wolff, *Stochastic Modeling and the Theory of Queues*. Englewood Cliffs, N.J.: Prentice Hall, 1989.

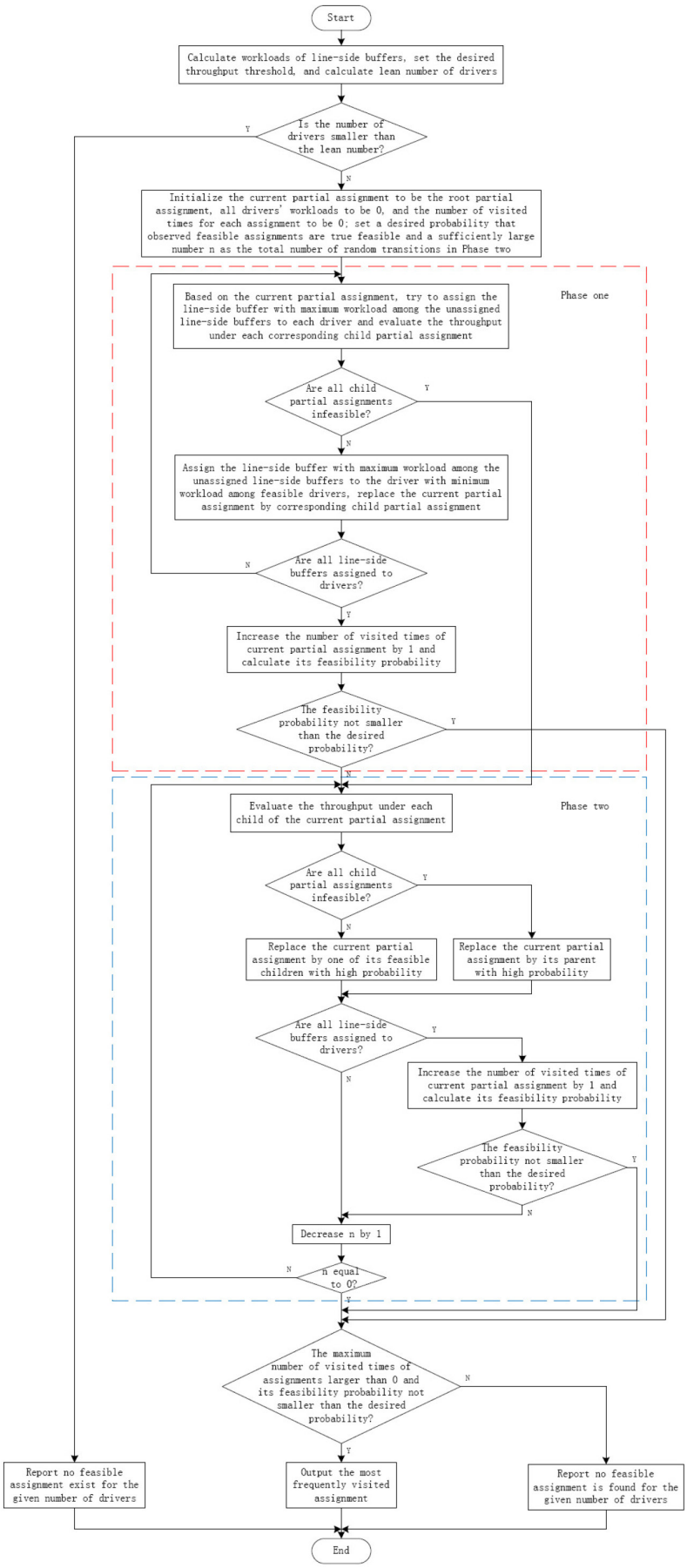
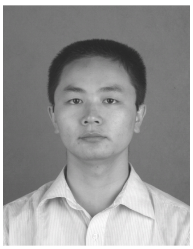
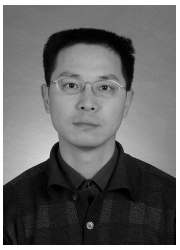


Fig. 9. The flowchart of the Sequential Assignment with Backtracking (SAB) algorithm



Chao-Bo Yan (S'06) received the B.S. degree from Xi'an Jiaotong University, Xi'an, China, in July 2004. He is pursuing the Ph.D. degree in the Center for Intelligent and Networked Systems (CFINS), Tsinghua University. He is currently a Visiting Scholar at the University of Michigan, Ann Arbor, MI, from October 2009.

His research interests include simulation of the Discrete Event Dynamic Systems (DEDS) and modeling, analysis, and optimization of production systems.

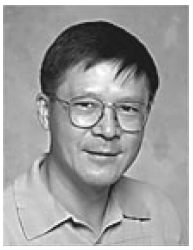


Qianchuan Zhao (M'06-SM'08) received the B.E. degree in automatic control in July 1992, the B.S. degree in applied mathematics in July 1992, and Ph.D. degree in control theory and its applications in July 1996, all from Tsinghua University, Beijing, China.

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. He is an associate editor for *Journal of Optimization Theory and Applications* and an associate editor for *IEEE Transactions on Automation Science and Engineering*.

Ningjian Huang received his Ph.D. degree in systems engineering from Oakland University in 1991.

He joined GM the same year, 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.



Guoxian Xiao received a bachelor's in mechanical engineering and a master's in manufacturing engineering from Northwestern University, China in 1982 and 1984, respectively. Then he served as an assistant professor at Northeastern University, Shenyang, China. He earned his doctorate in mechanical engineering at the University of Massachusetts at Amherst in 1995. He is a staff researcher at General Motors Research and Development Center. With GM since 1997, he leads several projects on the research and development of advanced technologies in machining process, real-time plant floor system, and re-manufacturing system.

His honors include five US patents, four GM's Boss Kettering Innovation Awards (2000, 2005, 2006, 2008), four Charles L. McCuen Special Achievement Awards (1998, 2005, 2006, 2008), John Campbell Award (2006), ASME Blackall Machine Tool and Gage Tool Award (2003), in machining process, production and maintenance system researches. He has authored more than 30 technical papers for journals and conferences. He currently serves on the Board of Steering Committee of the Society of Manufacturing Engineers' Machining and Material Removal Community, and associate editor for *IEEE Conference on Automation Science and Engineering*.

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

From 2000 to 2006, he was a Staff Research Engineer in Manufacturing Systems Research Lab, General Motors Research & Development Center, Warren, MI. He joined University of Kentucky as an Assistant Professor in Department of Electrical and Computer Engineering and Center for Manufacturing in 2006. To date he has published about 80 refereed journal and conference papers. He was also the Associate Editor of *IEEE Transactions on Automation Science and Engineering*, and *Mathematical Problems in Engineering*. He received the 2009 IIE Transactions - Design and Manufacturing Best Application Paper Award, 2005 IEEE Transactions on Automation Science and Engineering Best Paper Award, 2006 IEEE Early Industry/Government Career Award in Robotics and Automation, and he was also in 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. His primary research interests are in modeling, analysis and control of manufacturing, service, and health care systems.