Scheduling a Single-Arm Multi-Cluster Tool With a Condition-Based Cleaning Operation

Qinghua Zhu, *Senior Member, IEEE*, Hongpeng Li, Cong Wang, and Yan Hou

***Abstract*—As wafer circuit widths shrink less than 10 nm, stringent quality control is imposed on the wafer fabrication processes. Therefore, wafer residency time constraints and chamber cleaning operations are widely required in chemical vapor deposition, coating processes, etc. They increase scheduling complexity in cluster tools. In this paper, we focus on scheduling single-arm multi-cluster tools with chamber cleaning operations subject to wafer residency time constraints. When a chamber is being cleaned, it can be viewed as processing a virtual wafer. In this way, chamber cleaning operations can be performed while wafer residency time constraints for real wafers are not violated. Based on such a method, we present the necessary and sufficient conditions to analytically check whether a single-arm multi-cluster tool can be scheduled with a chamber cleaning operation and wafer residency time constraints. An algorithm is proposed to adjust the cycle time for a cleaning operation that lasts a long cleaning time. Meanwhile, algorithms for a feasible schedule are also derived. And an algorithm is presented for operating a multi-cluster tool back to a steady state after the cleaning. Illustrative examples are given to show the application and effectiveness of the proposed method.**

***Index Terms*—Chamber cleaning, multi-cluster tools, scheduling, semiconductor manufacturing.**

I. Introduction

CLUSTER tools (CTs) are widely used in multiple wafer fabrication processes such as etching, coating, and deposition. A typical CT consists of several processing chambers (PCs), a transportation robot, and two loadlocks (LLs) where wafer cassettes are loaded and unloaded. The robot is equipped with one or two arms, which are categorized into two kinds. One is a single-arm cluster tool, and the other is a dual-arm cluster tool. A single-arm robot can carry only one wafer, while a dual-arm robot can carry two. A cluster tool is so expensive that the equipment cost is more than half of the total investment in a modern semiconductor foundry. For higher throughput and more complex fabrication demands, two or more CTs are connected to form a multi-cluster tool system in which two adjacent CTs are linked by a shared buffer chamber (BC) with a capacity of one or two for holding incoming and outgoing wafers. A multi-cluster tool is formed by K individual CTs in a linear or tree-like topology, which is also called a K-CT. A linear single-arm multi−cluster tool (MCT) is illustrated in Fig. 1.

For a CT, the robot moving time between two PCs, or a PC and LLs, is much shorter than the wafer processing time at a PC in practice. It indicates that the system cycle time is decided by the time taken for completing a wafer at the bottleneck step. With such a property, it is said that a K-CT operates in a process-dominant mode. A backward strategy is optimal for a process-bound single-arm MCT during a steady state [1]−[3], while a swap strategy is efficient for a process-bound dual-arm CT [4], [5]. With these two scheduling strategies, extensive studies on scheduling CTs have been done [6]−[11].

Some wafer fabrication processes are subject to strict wafer residency time constraints (WRTCs). It means that a wafer must be removed from a PC within a limited time after being processed; otherwise, the wafer surface may be damaged by the residual chemicals, particles, and high temperature in a PC. Owing to considerations of such constraints and revisiting processes, the scheduling problem of a multi-cluster tool differs from the robotic manufacturing systems [12]−[15]. The robotic manufacturing system is configured in a linear layout with one input station and one output station; however, a multi-cluster tool is configured in a circular layout with the same input and output station, LLs. Due to the above differences, the methods in [12]−[14] cannot handle the deadlock problem for a multi-cluster tool.

Combined with WRTCs, extensive studies have been done on the modeling, scheduling, and performance evaluation of single-arm CTs [16]−[22] and multi-cluster tools [23]−[29]. A CT goes through three periods sequentially during the whole wafer fabrication: start-up, steady-state, and close-down ones. The first and last ones are also called transient states [30]−[37]. Most of the time, a CT operates in a steady state which is commonly seen in the semiconductor industry for its easy implementation. Note that a cleaning requirement is neglected in the above studies.

As wafer circuit widths shrink below 10 nanometers, extremely strict quality control is required for wafer fabrication processes. Residual gases and chemicals remain in a PC after a wafer is processed over, which can be accumulated inside the chamber. When such accumulations amount to a given threshold, they are detrimental to the quality of the next

Manuscript received August 28, 2022; accepted September 8, 2022. This work was supported in part by the Natural Science Foundation of Guangdong Province, China (2022A1515011310). Recommended by Associate Editor Derui Ding. *(Corresponding author: Qinghua Zhu.)*

Citation: Q. H. Zhu, H. P. Li, C. Wang, and Y. Hou, "Scheduling a single-arm multi-cluster tool with a condition-based cleaning operation," *IEEE/CAA J. Autom. Sinica*, vol. 10, no. 10, pp. 1965–1983, Oct. 2023.

Q. H. Zhu, H. P. Li, and Y. Hou are with the School of Computer Science and Technology, Guangdong University of Technology, Guangzhou 510006, China (e-mail: zhuqh@gdut.edu.cn; 2112005036@mail2.gdut.edu.cn; houyan @gdut.edu.cn).

C. Wang is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102-1982 USA (e-mail: wangcong@njit.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2023.123327

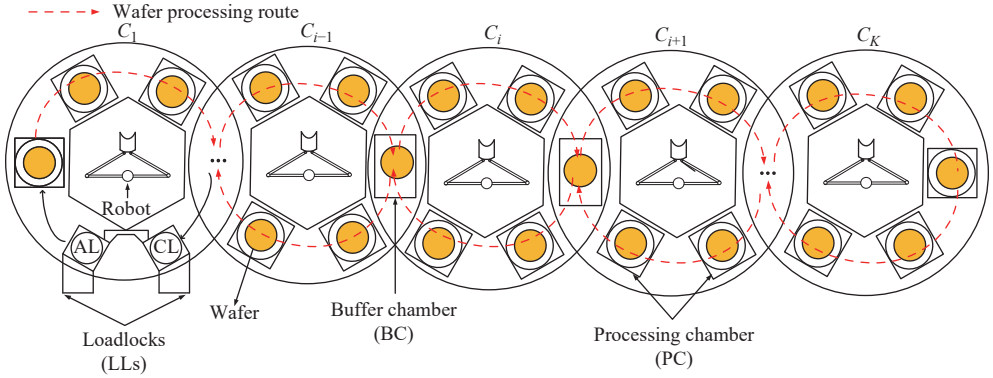

Fig. 1. A single-arm multi-cluster tool.

wafer to be processed at this chamber. Consequently, a chamber needs to be cleaned after it finishes processing d (also called cleaning period) wafers. Chamber cleaning with a period $d = 1$ is called purge, which is widely used in etching, chemical vapor deposition, and physical vapor deposition. With a purge operation, some methods are proposed for scheduling single-arm or dual-arm CTs. In [38] and [39], a search method is presented to find a globally optimal schedule for single-arm or dual-arm CTs with partially loading wafers into parallel PCs at a step. In addition, with a partially loading strategy, a linear programming-based method is proposed to check the schedulability of a single-arm CT with WRTCs. An algorithm is proposed to find a feasible schedule [40]. Although a purge operation can bring quality improvement for wafer fabrication, it decreases the system throughput due to frequent purge operations. In many scenarios, it is also vital to employ a general case that requires a chamber to be cleaned after processing a given number of wafers ($d > 1$) [41]. The work in [42] develops heuristic scheduling strategies and proposes methods that can synchronize the different cleaning periods for each step, which can meet more than one cleaning requirement in chambers. The residual heat and chemical gases in a chamber can be measured accurately with advanced sensor technologies. The work in [42] introduces a condition-based cleaning operation that is performed based on the real-time status of a chamber. This type of cleaning operation starts a cleaning procedure right after a chamber finishes processing a wafer if the controller has sent a cleaning signal. For a condition-based cleaning operation, a multi-agent reinforcement learning approach is derived to meet the cleaning requirements [43]. With WRTCs considered at the same time, [44] presents scheduling approaches by treating LLs as a temporary buffer to contain work-in-process (WIP) wafers to perform such a cleaning operation.

Combined with big data and machine learning technologies, the measured data of the residual heat and chemical gases in a chamber allows the controller to predict when a chamber should be cleaned. It means that we can know how many wafers will be consecutively processed before this chamber begins its cleaning operation. In such a case, the condition-based cleaning operation can be performed under a real-time prediction of a cleaning period, which indicates that a cleaning period is variable for a chamber. We focus on a process-dominant and time-constrained single-arm MCT with such a condition-based cleaning operation in this study. The work in [45] presents a virtual wafer method for such a cleaning operation in a single-arm CT with WRCTs.

The differences between this study and the aforementioned ones are summarized as follows.

1) Differently from [38]−[40], to meet stringent quality control and improve productivity gain, we consider a single-arm K-CT with a condition-based cleaning operation when a cleaning period $d > 1$ holds.

2) Compared to [42], [43], we focus on the parallel chambers and WRTCs in a multi-cluster tool instead of a single-cluster tool [42], [43]. In such a circumstance, we present a closed-form solution for schedulability analysis and a feasible schedule for a single-arm K-CT with a condition-based cleaning operation.

3) Differently from [44], [45], this work pays more attention to solving the scheduling problem with a condition-based cleaning operation in a multi-cluster tool instead of a single-cluster tool. Therefore, the methods in [44], [45] cannot solve such a scheduling problem. In particular, when a cleaning procedure lasts a long time, the method in [45] intends to load more virtual wafers until obtaining enough time for cleaning, which lowers productivity. We increase cycle time to gain enough time for cleaning without violating WRTCs, which indicates higher productivity. In addition, such a cleaning situation is ignored in [44]. Furthermore, compared with [44], no WIP wafers are staying in LLs during a cleaning operation, which improves productivity gain.

The existing methods in the aforementioned ones are no longer applicable when a condition-based cleaning operation is performed for a single-arm MCT with WRCTs and parallel chambers. Thus, this work aims to address such a new and challenging problem for a single-arm MCT with WRTCs and a condition-based cleaning operation.

This paper investigates the schedulability and scheduling problem for a single-arm MCT with a condition-based cleaning operation and WRTCs. We first illustrate two kinds of workloads of a single-arm MCT under a steady state with parallel chambers. Then, to meet a cleaning requirement, we extend a backward strategy by using a virtual-wafer approach. A virtual wafer does not interrupt a cleaning procedure in a chamber. Because such a wafer is fictitious, it implies that a robot just waits there idly until its next action. Thus, we need to find the right time to unload a virtual wafer from LLs in

order to load it into a chamber that is performing a cleaning operation.

This study makes the following contributions:

1) The necessary and sufficient conditions are presented to check whether a single-arm MCT is schedulable when a condition-based cleaning operation occurs; and

2) To meet WRTCs, algorithms are derived to find a feasible schedule if such a single-arm MCT is schedulable. An algorithm is proposed to switch a transient state to a steady state.

The remainder of this paper is organized as follows. In Section II, we first model robot activities and wafer processing time. In such a model, the workload analysis of a single-arm MCT during a steady state and a cleaning operation is presented in Section III. Then, for a condition-based cleaning operation that lasts a long time, the necessary and sufficient conditions for schedulability, and the scheduling algorithms are proposed in Section IV. Two examples illustrate the application and effectiveness of the proposed method in Section V. Finally, the conclusions are given in Section VI.

II. Modeling Activity Time

A. The Properties of a Single-Arm K-CT

A serial K-CT ($K \geq 2$) is composed of K single-arm cluster tools in a serial layout as shown in Fig. 1. Two adjacent tools are connected by a BC. Following [23], we have the assumptions below for a serial K-CT.

1) A K-CT operates in a steady state at most of the time because two LLs can work alternatively with no interruption such that wafers are always processed in chambers before a cleaning operation starts.

2) A BC without a processing function can only hold one wafer at a time.

3) Only one type of wafer is processed with the same processing route. A wafer is processed at a PC once and goes through a BC twice.

4) The time taken for the robot's moving, loading, unloading, and wafer processing at PCs is constant.

5) K individual tools are equipped with single-arm robots. Within each CT, a PC can process one wafer at a time.

Let $\mathrm{N}_l = \{0, 1, 2, 3, \ldots, l\}$ and $\mathrm{N}_l^+ = \mathrm{N}_l \setminus \{0\}$. In particular, define $\mathrm{N}_l = \emptyset$ if $l < 0$. Furthermore, as shown in Fig. 1, let C_i and R_i, $i \in \mathrm{N}_K^+$, denote the ith tool and its robot respectively. The head tool is denoted by C_1 with two LLs, while C_K is the tail tool without downstream peers. A buffer chamber between C_i and C_{i+1}, indexed by $b[i]$, is seen as an outgoing buffer for C_i and an incoming one for C_{i+1}, $i \in \mathrm{N}_{K-1}^+$. We assume that there are $n[i] + 1$ steps in C_i by treating the BC as a processing step with zero processing time. The wafer flow pattern in a tool can be defined as $(m_1^i, m_2^i, \ldots, m_j^i, \ldots, m_{n[i]}^i)$, where m_j^i denotes the number of parallel PCs at Step j in tool C_i, $i \in \mathrm{N}_K^+$, $j \in \mathrm{N}_{n[i]}^+$. Specifically, $m_{b[i]}^i = 1$, $i \in \mathrm{N}_{K-1}^+$, which implies that there is one buffer chamber. The steps in C_i are denoted by $\mathrm{PS}_{i,0}$, $\mathrm{PS}_{i,1}$, …, and $\mathrm{PS}_{i,n[i]}$, where $\mathrm{PS}_{i,0}$ and $\mathrm{PS}_{i,b[i]}$ are the incoming and outgoing steps, respectively. As shown in Fig. 1, we can know the wafer processing route is $\mathrm{PS}_{1,0} \rightarrow \mathrm{PS}_{1,1} \rightarrow \cdots \rightarrow \mathrm{PS}_{1,b[1]}$ ($\mathrm{PS}_{2,0}$) $\rightarrow \mathrm{PS}_{2,1} \rightarrow \cdots \rightarrow \mathrm{PS}_{2,b[2]}$ ($\mathrm{PS}_{3,0}$) $\rightarrow \cdots \rightarrow \mathrm{PS}_{K-1,b[K-1]}$ ($\mathrm{PS}_{K,0}$) $\rightarrow \mathrm{PS}_{K,1} \rightarrow \cdots \rightarrow \mathrm{PS}_{K,n[K]} \rightarrow \cdots \rightarrow \mathrm{PS}_{K,0}$ ($\mathrm{PS}_{K-1,b[K-1]}$) $\rightarrow \mathrm{PS}_{K-1,b[K-1]+1} \rightarrow \cdots \mathrm{PS}_{3,0}$ ($\mathrm{PS}_{2,b[2]}$) $\rightarrow \mathrm{PS}_{2,b[2]+1} \rightarrow \cdots \mathrm{PS}_{2,0}$ ($\mathrm{PS}_{1,b[1]}$) $\rightarrow \cdots \rightarrow \mathrm{PS}_{1,n[1]} \rightarrow \mathrm{PS}_{1,0}$. Then, $\mathrm{PC}_{k,j}^i$ denotes the kth chamber at $\mathrm{PS}_{i,j}$, $k \in \mathrm{N}_{m_j}^+$, $i \in \mathrm{N}_K^+$, $j \in \mathrm{N}_{n[i]}^+$. The robot waiting time before unloading a wafer and the processing time at $\mathrm{PS}_{i,j}$, $j \in \mathrm{N}_{n[i]}^+$ are denoted by $\omega_{i,j}$ and $\alpha_{i,j}$, respectively. Due to no processing function in BC and LLs, the processing time of these steps is zero. μ_i denotes the time taken by robot R_i's moving between two PCs or a PC and LLs, and λ_i denotes the time taken for robot R_i's loading/unloading a wafer into/from a PC. Let $S_{\mathrm{up}}^i = \sum_{j=1}^{b[i]-1} m_j^i$, $i \in \mathrm{N}_{K-1}^+$ and $S_{\mathrm{up}}^0 = 0$. Then, we let $S_{\mathrm{down}}^K = \sum_{q=1}^{K-1} S_{\mathrm{up}}^q + \sum_{j=1}^{n[K]} m_j^K$. Specifically, $S_{\mathrm{down}}^K = S_{\mathrm{down}}^{K-1}$. S_{down}^w can be calculated by

$$S_{\mathrm{down}}^w = S_{\mathrm{down}}^{w+1} + \sum_{j=b[w]+1}^{n[w]} m_j^w \tag{1}$$

where $2 \leq w \leq K-1$. An example for (1) is shown in Fig. 2.

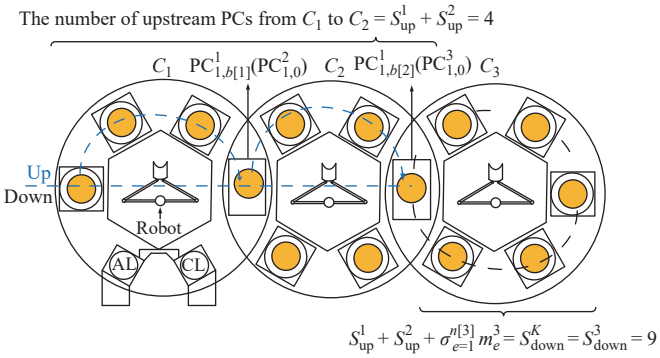

Fig. 2. Illustration of S_{up}^i and S_{down}^K.

B. The Properties of Each Single-Arm CT

The key to scheduling a multi-cluster tool is to coordinate the actions of the multiple robots to satisfy WRCTs and cleaning operation requirements. To do so, we need to analyze time properties.

During a steady state, a backward strategy is optimal for a process-dominant K-cluster tool where the robot in each CT performs the following operation sequence.

$\Gamma = \langle$Moving to $\mathrm{PS}_{i,n[i]}$ (μ_i) $\rightarrow$ waiting there for $\omega_{i,n[i]}$ time units $\rightarrow$ unloading a finished wafer from the $m_{n[i]}^i$th chamber ($\mathrm{PC}_{m_{n[i]},n[i]}^i$) at $\mathrm{PS}_{i,n[i]}$ (λ_i) $\rightarrow$ moving to LLs and loading it there ($\mu_i + \lambda_i$) $\rightarrow$ moving to $\mathrm{PS}_{i,n[i]-1}$ (μ_i) $\rightarrow$ waiting there for $\omega_{i,n[i]-1}$ time units $\rightarrow$ unloading a completed wafer from its $m_{n[i]-1}^i$th chamber ($\mathrm{PC}_{m_{n[i]-1},n[i]-1}^i$) at $\mathrm{PS}_{i,n[i]-1}$ (λ_i) $\rightarrow$ moving to $\mathrm{PS}_{i,n[i]}$ and loading it into the $m_{n[i]}^i$th chamber ($\mathrm{PC}_{m_{n[i]},n[i]}^i$) at $\mathrm{PS}_{i,n[i]}$ ($\mu_i + \lambda_i$) $\rightarrow$ moving to $\mathrm{PS}_{i,n[i]-2}$ (μ_i) $\rightarrow$ waiting there for $\omega_{i,n[i]-2}$ time units $\rightarrow$ unloading a completed wafer from its $m_{n[i]-2}^i$th chamber ($\mathrm{PC}_{m_{n[i]-2},n[i]-2}^i$) at $\mathrm{PS}_{i,n[i]-2}$ (λ_i) $\rightarrow \cdots \rightarrow$ moving to $\mathrm{PS}_{1,1}$ and loading a wafer into the m_1^ith chamber ($\mathrm{PC}_{m_1,1}^i$) at $\mathrm{PS}_{i,1}$ ($\mu_i + \lambda_i$) $\rightarrow$ moving to $\mathrm{PS}_{i,n[i]}$ (μ_i) again $\rightarrow$ waiting there for $\omega_{i,n[i]}$ time units $\rightarrow$ unloading a finished

TABLE I
SYMBOLS AND ASSOCIATED TIME FOR A SINGLE-ARM MCT

Symbol	Meaning
λ_i	Time spent by a robot to finish loading/unloading a wafer into/from a PC, $i \in \mathrm{N}_K^+$
μ_i	Robot moving time between a PC/LLs and a PC, $i \in \mathrm{N}_K^+$
$\alpha_{i,j}$	Wafer processing time at $\mathrm{PS}_{i,j}$, $i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+$
$\tau_{i,j}$	The wafer sojourn time at $\mathrm{PS}_{i,j}$, $i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+$
$\delta_{i,j}$	The allowed longest time for a wafer to stay a PC of $\mathrm{PS}_{i,j}$, $i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+$
$\omega_{i,j}$	Robot's waiting time before unloading a wafer of $\mathrm{PS}_{i,j}$, $i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+$
$\rho_{i,j}$	The taken time of a condition-based cleaning operation at a chamber of $\mathrm{PS}_{i,j}$, $i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+$
Θ	The cycle time of a single-arm MCT

wafer from its $(m_{n[i]}^i - 1)$th chamber ($\mathrm{PC}_{m_{n[i]}^i - 1, n[i]}^i$) at $\mathrm{PS}_{i,n[i]}$ (λ_i) … ⟩.

Therefore, robot R_i's cycle time is

$$\psi_i = 2(n[i]+1)(\mu_i + \lambda_i) + \sum_{j=0}^{n[i]} \omega_{i,j} = \psi_{i,1} + \psi_{i,2},\ i \in \mathrm{N}_K^+ \tag{2}$$

where the robot task time $\psi_{i,1} = 2(n[i] + 1)(\mu_i + \lambda_i)$ is a constant, while $\psi_{i,2} = \sum_{j=0}^{n[i]} \omega_{i,j}$ is its waiting time during a cycle.

At $\mathrm{PS}_{i,j}$, the lower bound time required to complete a wafer in a cycle is [18]

$$\Pi_{i,jL} = \frac{\alpha_{i,j} + 4\lambda_i + 3\mu_i}{m_j^i},\ i \in \mathrm{N}_K^+,\ j \in \mathrm{N}_{n[i]}^+ \tag{3}$$

and the upper bound time required to complete a wafer in a cycle is [18]

$$\Pi_{i,jU} = \frac{\alpha_{i,j} + \delta_{i,j} + 4\lambda_i + 3\mu_i}{m_j^i},\ i \in \mathrm{N}_K^+,\ j \in \mathrm{N}_{n[i]}^+. \tag{4}$$

Let $\tau_{i,j}$ denote a wafer sojourn time at $\mathrm{PS}_{i,j}$, $i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+$. If WRTCs at $\mathrm{PS}_{i,j}$ are satisfied, we have $\alpha_{i,j} \le \tau_{i,j} \le \alpha_{i,j} + \delta_{i,j}$. Therefore, a feasible schedule is defined as follows.

Definition 1: Given WRTCs time interval $[\alpha_{i,j},\ \alpha_{i,j} + \delta_{i,j}]$ at $\mathrm{PS}_{i,j}$, $i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+ \setminus \{b[i]\}$, a schedule is feasible if $\alpha_{i,j} \le \tau_{i,j} \le \alpha_{i,j} + \delta_{i,j}$ always holds for each CT.

The cycle time at $\mathrm{PS}_{i,j}$ is [18]

$$\theta_{i,j} = \frac{\tau_{i,j} + 4\lambda_i + 3\mu_i + \omega_{i,j-1}}{m_j^i},\ i \in \mathrm{N}_K^+,\ j \in \mathrm{N}_{n[i]}^+ \tag{5}$$

and

$$\theta_{i,0} = \tau_{i,0} + 4\lambda_i + 3\mu_i + \omega_{i,n[i]},\ i \in \mathrm{N}_K^+. \tag{6}$$

By (5) and (6), for a time-constrained single-arm MCT, the key to finding a feasible schedule is how to allocate robot waiting time $\omega_{i,j}$. Let $\Pi_i = \max\{\Pi_{i,0L}, \Pi_{i,1L}, \ldots, \Pi_{i,n[i]L}, \psi_{i,1}\}$. If $\Pi_i = \psi_{i,1}$, $i \in \mathrm{N}_K^+$, a single-arm MCT is transport-dominant, and otherwise it is process-dominant. During a steady state, let Θ_i denote the cycle time of C_i. To meet WRTCs, a process-bound CT should be scheduled such that

$$\Theta_i = \theta_{i,j} = \theta_{i,0} = \psi_i \text{ and } \tau_{i,j} \in [\alpha_{i,j}, \alpha_{i,j} + \delta_{i,j}],\ j \in \mathrm{N}_{n[i]}^+. \tag{7}$$

Therefore, $\tau_{i,j}$ can be calculated as follows.

$$\tau_{i,j} = \psi_i \times m_j^i - (4\lambda_i + 3\mu_i + \omega_{i,j-1}),\ i \in \mathrm{N}_K^+,\ j \in \mathrm{N}_{n[i]}^+ \tag{8}$$

$$\tau_{i,0} = \psi_i - (4\lambda_i + 3\mu_i + \omega_{i,n[i]}),\ i \in \mathrm{N}_K^+. \tag{9}$$

The cycle time of a K-CT is denoted by Θ, and let $\Pi = \max\{\Pi_1, \Pi_2, \ldots, \Pi_K\}$. We presume $\Pi_h = \Pi$ with $h \in \mathrm{N}_K^+$, which implies that Π_h has the heaviest workload among K CTs. According to [23], to obtain a cyclic schedule for a process-dominant K-cluster tool, we have

$$\Theta = \Theta_1 = \Theta_2 = \cdots = \Theta_K \tag{10}$$

where $\Theta \ge \Pi = \Pi_h$. By (10), due to Π being the lower-bound cycle time, each CT is scheduled with the same cycle time of a K-cluster tool, which means $\Theta \ge \Pi = \Pi_h$ holds for every individual CT. Thus, it is justified to apply a backward strategy to every individual tool. The above symbols and associated time for a single-arm MCT are summarized in Table I.

During a steady state, to obtain a feasible cyclic schedule for a process-dominant K-CT with WRTCs, we have the following result from [23].

Given $\Theta \ge \Pi = \Pi_h$ as the cycle time for a process-dominant K-cluster tool with WRTCs, a feasible cyclic schedule exists, if and only if for C_i and C_{i+1}, $i \in \mathrm{N}_{K-1}^+$, $\omega_{i,j}$ and $\omega_{(i+1),f}$ are set such that the following conditions are satisfied:

$$\Theta = \theta_{i,j} = \theta_{(i+1),f},\ j \in \mathrm{N}_{n[i]} \text{ and } f \in \mathrm{N}_{n[i+1]} \tag{11}$$

$$\tau_{i,j} \in [\alpha_{i,j}, \alpha_{i,j} + \delta_{i,j}],\ j \in \mathrm{N}_{n[i]} \setminus \{0, b[i]\} \tag{12}$$

$$\tau_{i,b[i]} \ge 4\lambda_{i+1} + 3\mu_{i+1} + \omega_{(i+1),n[i+1]} \tag{13}$$

and

$$\tau_{i+1,0} \ge 4\lambda_i + 3\mu_i + \omega_{i,b[i]-1}. \tag{14}$$

In the above conditions, (11), (13) and (14) are related to the BCs for connecting the adjacent CTs, while (12) constrains every processing step in a K-CT. Therefore, to schedule a K-CT, the key is to make each robot in CT operate in a paced way when accessing a wafer in a BC and regulate the wafer sojourn time for every step to satisfy WRTCs. In a steady state, we suppose that $\tau_{i,b[i]} = 4\lambda_{i+1} + 3\mu_{i+1} + \omega_{(i+1),n[i+1]}$ holds to let robot R_{i+1} unload a wafer immediately from a buffer after it is loaded by R_i.

With the above analysis, under a steady state, a scheduling method for a single-arm MCT with parallel chambers is proposed next.

III. Schedulability and Scheduling Algorithm

A K-CT mostly runs in a steady state when no cleaning operation is required to perform. When a condition-based cleaning operation is performed, a K-CT enters a transient state. Thus, we first ensure that a K-CT is schedulable during a steady state, and then focus on its schedulability with a condition-based cleaning operation during a transient state.

A. Schedulability During a Steady State

For a process-dominant K-CT with no WRTCs, Zhu *et al.* [23] find a one-wafer cyclic schedule and propose an efficient method to obtain the optimal schedule with the minimal cycle time Θ. To schedule a single-arm CT with cycle time Θ , the key is to determine when a robot should wait and how long it should wait, so is for a K-CT. We propose Algorithm 3 to decide the robot waiting time $\omega_{i,j}$, $i \in \mathrm{N}_K^+$, $j \in \mathrm{N}_{n[i]}$, and then obtain a schedule for each CT. When every CT is scheduled, a feasible schedule is obtained for a K-CT during a steady state. There are two cases in scheduling a process-bound single-arm CT.

Case 1: If $\psi_{i,1} \le \Pi$ and $\Theta \le \Pi_{i,jU}$, $i \in \mathrm{N}_K^+$, $j \in \mathrm{N}_{n[i]}^+$, hold, the workloads of all processing steps within C_i are balanced and C_i is process-bound.

Case 2: If $\cap_{j \in \mathrm{N}_{n[i]} \setminus \{0, b[i]\}} [\Pi_{i,jL}, \Pi_{i,jU}] = \emptyset$, $i \in \mathrm{N}_K^+$, holds, then the workloads of some processing steps within C_i are unbalanced.

For Case 1, we set the robot waiting time $\omega_{i,j}$ via (15) in Algorithm 1. For the unbalanced workloads like Case 2, we set $\omega_{i,j}$ via (16) in Algorithm 2.

$$\omega_{i,j} = \min\{\Theta - \Pi_{i,(j+1)L}, \Theta - \psi_{i,1} - \omega_{i,(n[i])} - \sum_{e \in \mathrm{N}_{j-1}} \omega_{i,e}\} \tag{15}$$

$$\omega_{i,j-1} = \begin{cases} \min\{\Theta - \Pi_{i,jL}, \Theta - \psi_{i,1} - \sum\limits_{e \in \mathrm{N}_{j-1} \cup \{n[i]\}} \omega_{i,e}\}, & j \in F_i \\ (\Theta - \Pi_{i,jU}) \times m_j^i, & j \in E_i. \end{cases} \tag{16}$$

Algorithm 1 Set the Robot Waiting Time for the Case of Balanced Workloads (Case 1)

Input: $\alpha_{i,j}$, λ_i, μ_i, $\Pi_{i,jL}$ $(i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+)$, Θ
Output: $\omega_{i,j}$

Initialization:
  $\omega_{i,j} \leftarrow 0$, for each $j \in \mathrm{N}_{n[i]}^+$
  $\psi_{i,1} \leftarrow 2(n[i]+1)(\mu_i + \lambda_i)$, for each $i \in \mathrm{N}_K^+$
If $i = 1$ **then**
  /* set $\omega_{1,j}$, $j \in \mathrm{N}_{n[1]}$ */
  For $j \leftarrow 0$ **to** $n[i]-1$ **do**
    $\omega_{1,j} \leftarrow 0$
  $\omega_{1,n[1]} \leftarrow \Theta - \psi_{i,1}$
  $\tau_{1,b[1]} \leftarrow \Theta - (4\lambda_1 + 3\mu_1)$
If $i \ge 2$, $i \in \mathrm{N}_{K-1}^+$ **then**
  /* set $\omega_{i,j}$, $j \in \mathrm{N}_{n[i]}$ */
  If $i \le K-1$ **and** $\tau_{(i-1),b[i-1]} \ge 4\lambda_i + 3\mu_i$ **then**
    $\omega_{i,n[i]} \leftarrow \min\{\tau_{(i-1),b[i-1]} - 4\lambda_i + 3\mu_i, \Theta - \psi_{i,1}\}$
    For $j \leftarrow 0$ **to** $n[i]-1$ **and** $j \ne b[i]-1$ **do**
      $\omega_{i,j} \leftarrow \min\{\Theta - \Pi_{i,(j+1)L}, \Theta - \psi_{i,1} - \omega_{i,n[i]} - \sum_{e \in \mathrm{N}_{j-1}} \omega_{i,e}\}$
    $\omega_{i,b[i]-1} \leftarrow \Theta - \psi_{i,1} - \sum_{j \in \mathrm{N}_{n[i]} \setminus \{b[i]-1\}} \omega_{i,j}$
    $\tau_{i,b[i]} \leftarrow \Theta - (4\lambda_i + 3\mu_i + \omega_{i,b[i]-1})$
If $i = K$ **then**
  /* set $\omega_{K,j}$, $j \in \mathrm{N}_{n[K]}$ */
  For $j \leftarrow 0$ **to** $n[K]-1$ **do**
    $\omega_{K,j} \leftarrow \min\{\Theta - \Pi_{K,(j+1)L}, \Theta - \psi_{K,1} - \sum_{e \in \mathrm{N}_{j-1}} \omega_{K,e}\}$
  $\omega_{K,n[K]} \leftarrow \Theta - \psi_{K,1} - \sum_{e=0}^{n[K]-1} \omega_{K,e}$

Algorithm 2 Set the Robot Waiting Time for the Case of Unbalanced Workloads (Case 2)

Input: $\alpha_{i,j}$, λ_i, μ_i, $\Pi_{i,jL}$, $\Pi_{i,jU}$ $(i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]})$, m_j^i $(i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]}^+)$, Θ
Output: $\omega_{i,j}$, Q

Initialization:
  $Q \leftarrow 0$
  $\omega_{i,j} \leftarrow 0$, for each $j \in \mathrm{N}_{n[i]}^+$
  $E_i \leftarrow \{j \mid j \in \mathrm{N}_{n[i]}^+ \setminus b[i], \Pi_{i,jU} < \Theta\}$
  $\psi_{i,1} \leftarrow 2(n[i]+1)(\mu_i + \lambda_i)$, for each $i \in \mathrm{N}_K^+$
Set $\omega_{i,(j-1)}$, $j \in E_i$, as
  $\omega_{i,(j-1)} \leftarrow (\Theta - \Pi_{i,jU}) \times m_j^i$
  $\xi \leftarrow \Theta - \psi_{i,1} - \sum_{j \in E_i} \omega_{i,(j-1)}$
If $i = 1$ **then**
  /* set $\omega_{i,(j-1)}$, $j \in \mathrm{N}_{n[1]} \setminus E_i$ */
  If $\xi < 0$ **then**
    $Q \leftarrow -1$
  Else
    For $j \leftarrow 0$ **to** $n[i]-1$ **do**
      $\omega_{1,j} \leftarrow 0$
    $\omega_{1,n[1]} \leftarrow \xi$
    $\tau_{1,b[1]} \leftarrow \Theta - (4\lambda_1 + 3\mu_1)$
Else
  /* If $i \ge 2$, $i \in \mathrm{N}_K^+$, set $\omega_{i,j}$, $j \in \mathrm{N}_{n[i]} \setminus E_i$ */
  If $\xi < 0$ **then**
    $Q \leftarrow -1$
  Else
    If $i \le K-1$ **and** $\tau_{(i-1),b[i-1]} \ge 4\lambda_i + 3\mu_i$ **then**
      $\omega_{i,n[i]} \leftarrow \min\{\tau_{(i-1),b[i-1]} - (4\lambda_i + 3\mu_i), \xi\}$
      $\xi \leftarrow \xi - \omega_{i,n[i]}$
      For $j \in \mathrm{N}_{n[i]} \setminus \{b[i], E_i\}$ **do**
        $\omega_{i,(j-1)} \leftarrow \min\{\Theta - \Pi_{i,jL}, \xi\}$
        $\xi \leftarrow \xi - \omega_{i,(j-1)}$
      $\omega_{i,b[i]-1} \leftarrow \xi$
      $\tau_{i,b[i]} \leftarrow \Theta - (4\lambda_i + 3\mu_i + \omega_{i,b[i]-1})$
    Else
      For $j \in \mathrm{N}_{n[K]} \setminus E_K$ **do**
        $\omega_{K,(j-1)} \leftarrow \min\{\Theta - \Pi_{K,jL}, \xi\}$
        $\xi \leftarrow \xi - \omega_{K,(j-1)}$
      $\omega_{K,n[K]} \leftarrow \xi$

Theorem 1: If a process-bound CT, with the cycle time being Θ, satisfies the conditions in Case 1, then Algorithm 1 can find a feasible schedule for this CT.

Proof: The workloads of all processing steps are balanced, which implies $\psi_{i,1} \le \Theta$ and $\Theta \le \Pi_{i,jU}$, $j \in \mathrm{N}_{n[i]}^+$, hold for C_i. Then, a feasible schedule is obtained by setting the robot wait-

ing time via (15) in Algorithm 1.

i) By (15), if $\Theta - \Pi_{i,jL} \le \Pi - \psi_{i,1} - \sum_{e\in N_{j-1}\cup\{n[i]\}}\omega_{ie}$, then $\omega_{i,j} = \Theta - \Pi_{i,(j+1)L}$, the wafer sojourn time at $PS_{i,(j+1)}$ is $\tau_{i,(j+1)} = \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + \omega_{i,j}) = \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + (\Pi - \Pi_{i,(j+1)L}))$. Obviously, $\Theta - \Pi_{i,(j+1)L} \le (\Theta - \Pi_{i,(j+1)L}) \times m^i_{j+1}$. Thus, $\tau_{i,(j+1)} = \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + (\Theta - \Pi_{i,(j+1)L})) \ge \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + (\Theta - \Pi_{i,(j+1)L}) \times m^i_{j+1}) = \alpha_{i,(j+1)}$. Due to $\Theta \le \Pi_{i,jU}$, we have $\tau_{i,(j+1)} = \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + (\Theta - \Pi_{i,(j+1)L})) \le \Pi_{i,jU} \times m^i_{j+1} - (4\lambda_i + 3\mu_i) = \alpha_{i,(j+1)} + \delta_{i,(j+1)}$. Therefore, a feasible schedule is obtained for such a CT.

ii) By (15), if $\omega_{i,j} = \Theta - \psi_{i,1} - \sum_{e\in N_{j-1}\cup\{n[i]\}}\omega_{ie}$, which implies $\Theta - \Pi_{i,(j+1)L} > \Theta - \psi_{i,1} - \sum_{e\in N_{j-1}\cup\{n[i]\}}\omega_{ie}$. Similarly, $\tau_{i,(j+1)} = \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + (\Theta - \psi_{i,1} - \sum_{e\in N_{j-1}\cup\{n[i]\}}\omega_{ie})) > \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + (\Theta - \Pi_{i,(j+1)L})) \ge \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + (\Theta - \Pi_{i,(j+1)L}) \times m^i_{j+1}) = \alpha_{i,(j+1)}$. Similarly, due to $\Theta \le \Pi_{i,jU}$, we have $\tau_{i,(j+1)} = \Theta \times m^i_{j+1} - (4\lambda_i + 3\mu_i + (\Theta - \psi_{i,1} - \sum_{e\in N_{j-1}\cup\{n[i]\}}\omega_{ie})) \le \Pi_{i,jU} \times m^i_{j+1} - (4\lambda_i + 3\mu_i) = \alpha_{i,(j+1)} + \delta_{i,(j+1)}$. Therefore, by the above setting, a wafer can be completely processed while WRTCs for each step are satisfied. A feasible schedule is obtained for this CT. ■

In Algorithm 1, if $i = 1$, the number of iterations in the For-loop of Lines 4 and 5 is no more than $n[1]$. If $2 \le i \le K$, the number of iterations in the For-loop of Lines 11−12 and 16−17 is no more than $(n[i])^2$. $n[i]$ is a constant. Therefore, the computation complexity of Algorithm 1 is polynomial.

Theorem 2: A process-bound individual CT with Θ being the cycle time satisfies the conditions in Case 2, if Algorithm 2 returns $Q = 0$, a feasible schedule can be found for this CT.

Proof: For unbalanced workload steps, let $E_i = \{j | j \in N^+_{n[i]}\backslash\{b[i]\}, \Pi_{i,jU} < \Pi\}$, while $F_i = N^+_{n[i]} \backslash E_i$. F_i is the set of balanced workload steps. If Algorithm 2 returns $Q = 0$, it implies that the condition $\xi < 0$ in Lines 9 and 17 does not hold. In such a case, a feasible schedule is obtained by setting robot waiting time via (16) when $j \in F_i$. Similarly, it follows from Theorem 1 that WRTCs for Step j, $j \in F_i$, are all satisfied. For unbalanced workload steps, by setting $\omega_{i,(j-1)}$ via (16), $\tau_{i,j} = \Theta \times m^i_j - (4\lambda_i + 3\mu_i + \omega_{i,j-1}) = \Theta \times m^i_j - (4\lambda_i + 3\mu_i + (\Theta - \Pi_{i,jU}) \times m^i_j) = \alpha_{i,j} + \delta_{i,j}$. This means that WRTCs for any Step $j \in E_i$ are also satisfied. Therefore, WRTCs are satisfied for any Step j, $j \in F_i \cup E_i = N^+_{n[i]}$, in this CT. Thus, if Algorithm 2 returns $Q = 0$, a feasible schedule can be found for this CT. ■

In Algorithm 2, the number of iterations in For-loop, i.e., Lines 13 and 14, 24−26, and 30−32, is no more than $n[i]$. Thus, the computational complexity of Algorithm 2 is polynomial.

Within a process-dominant K-CT, each component CT is process-bound with balanced or unbalanced workloads. Therefore, we need to combine Algorithms 1 and 2 to obtain a schedule for a K-CT. For a process-dominant and time-constrained K-CT with parallel chambers, the above analysis can be summarized as the following Algorithm 3 to check its schedulability and calculate robot waiting time during a steady state.

Algorithm 3 Calculate the Robot Waiting Time for a K-CT During a Steady State

1: $Q \leftarrow 0$;
2: For $i \in N^+_K$, do:
2.1: If C_i satisfies the conditions in Case 1, then Algorithm 1 is called to set R_i's waiting time;
2.2: If C_i satisfies the conditions in Case 2, then Algorithm 2 is called to set R_i's waiting time and Q.
3: End

If Algorithm 3 returns $Q = 0$, it finds a feasible schedule for a K-CT by setting the robot waiting time. If $Q > 0$ is returned, a K-CT is unschedulable.

In Algorithm 3, the number of iterations in the For-loop is no more than a constant K. By Algorithm 3, if every CT within a K-CT runs under a balanced workload, then, in this worst case, the number of calculations of $\omega_{i,j}$ is $(n[i])^2 \times K$. Obviously, the computational complexity of Algorithm 3 is polynomial.

B. Scheduling Analysis of a Transient State

Without loss of generality, we assume that a condition-based cleaning operation is performed at $PS_{i,j}$, $i \in N^+_K$, $j \in N^+_{n[i]}\backslash\{b[i]\}$. In such a case, the cleaning operation starts when a wafer in the chamber of Step j is unloaded. For this cleaning operation, the following robot action sequence is performed in a backward strategy.

$\Gamma_1 = \langle$Moving the processed wafer from the chamber at $PS_{i,j}$ to the chamber at $PS_{i,(j+1)}$ $(\mu_i) \rightarrow$ loading the wafer there $(\lambda_i) \rightarrow$ moving to $PS_{i,(j-1)}$ $(\mu_i) \rightarrow$ waiting there for $\omega_{i,(j-1)}$ time units $\rightarrow$ unloading the processed wafer from the chamber at $PS_{i,(j-1)}$ $(\lambda_i) \rightarrow$ loading this wafer into the chamber at $PS_{i,j}$ $(\mu_i)\rangle$.

It takes $(2\lambda_i + 3\mu_i + \omega_{i,(j-1)})$ time units to complete the above operation sequence. Let $\rho_{i,j}$ denote the consumed time for a cleaning operation at $PS_{i,j}$, $i \in N^+_K$, $j \in N^+_{n[i]}\backslash\{b[i]\}$. Obviously, if $\rho_{i,j} \le 2\lambda_i + 3\mu_i + \omega_{i,(j-1)}$ holds, such a cleaning operation does not undermine the backward strategy, which implies that the CT can still run under a steady state. However, the consumed time of a condition-based cleaning operation tends to be much longer than $(2\lambda_i + 3\mu_i + \omega_{i,(j-1)})$ in practice. Thus, we presume that $\rho_{i,j} > 2\lambda_i + 3\mu_i + \omega_{i,(j-1)}$ always holds in this work. In such a case, when a cleaning operation is performed in a chamber, it is equivalent that the chamber spends the same time processing an imaginary wafer, called a virtual wafer. To keep the robot action sequence unchanged and facilitate finding a feasible schedule, we intend to load a virtual wafer into this chamber. Consequently, this chamber can perform the cleaning operation instead of processing because the loaded wafer is virtual. We have to determine when to unload a virtual wafer from LLs and load it into the chamber that is planned to be cleaned. In this work, we consider the case that only one condition-based cleaning operation occurs, i.e., only one condition-based cleaning operation is performed at the same time. With this assumption, we show how to determine when to load a virtual wafer before a cleaning operation begins.

To facilitate the presentation, it is supposed that the time when the robot completes its loading operation at Step 2 is the

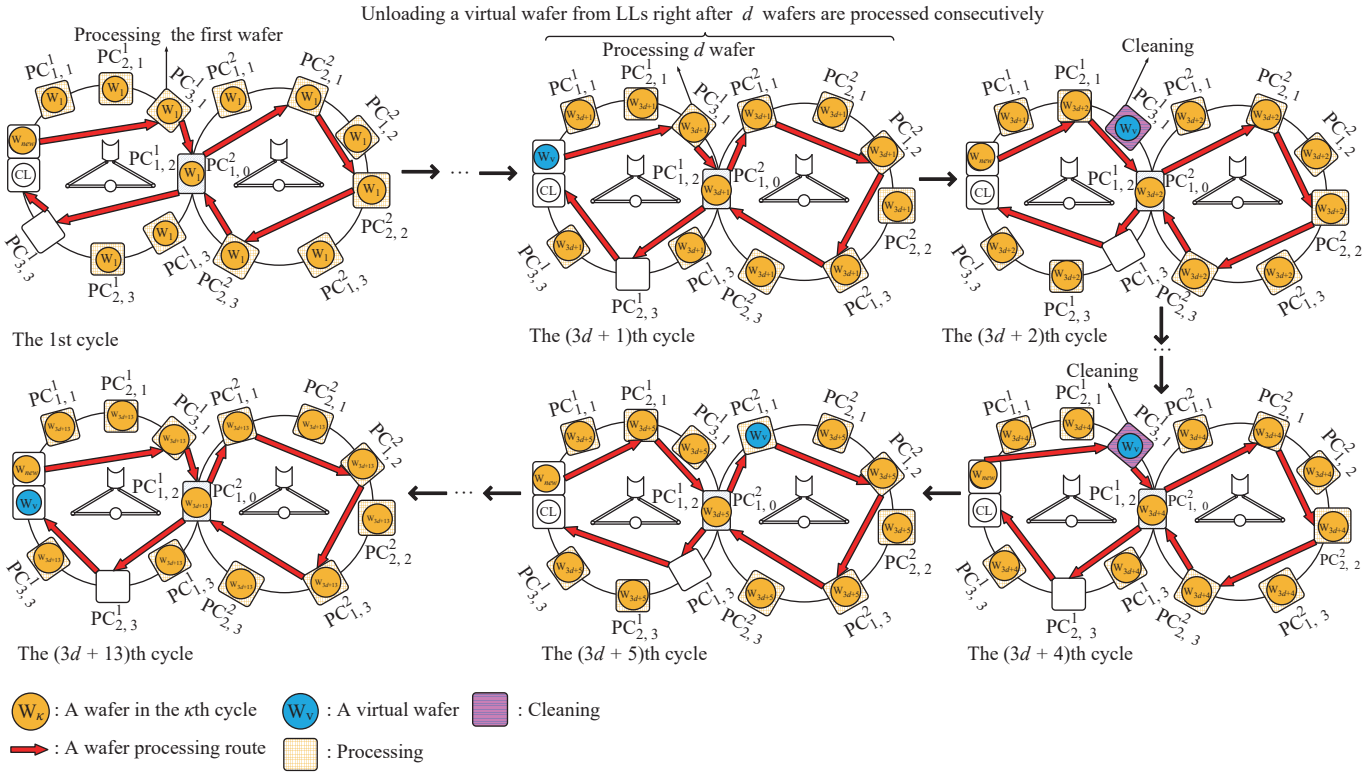

Fig. 3. Example for Situation 1. The condition-based cleaning operation occurs at $PC^1_{3,1}$ after d wafers are processed consecutively.

beginning of each cycle, which implies that the robot is about to move to LLs. The time point when a virtual wafer is unloaded from LLs is denoted by t. The chamber at $PS_{i,j}$ should be cleaned after processing d wafers consecutively, which can be known from the controller. With such a circumstance, the datum of t is the beginning of the cycle that this chamber has processed its first wafer, which is the first cycle. As the condition-based cleaning operation occurs at $PS_{i,j}$, depending on where the chamber requiring a cleaning operation is, we have two situations.

Situation 1: A condition-based cleaning operation occurs at $PS_{1,1}$.

Situation 2: A condition-based cleaning operation occurs at $PS_{i,j}$, $i \in N^+_K, j \in N^+_{n[i]} \backslash \{b[i]\}, j \neq 1$ if $i = 1$.

For Situation 1, the time to unload a virtual wafer from LLs is right after a chamber at $PS_{1,1}$ consecutively processes d wafers. For easy presentation, we presume that C_1 is set up with a wafer flow pattern (m^1_1, m^1_2) and such a cleaning operation occurs at $PS_{1,1}$ when this chamber at $PS_{1,1}$ has processed d wafers in a row. Then, when the time comes to the $(m^1_1 \times d + 1)$th cycle or time point $(m^1_1 \times d + 1) \times \Theta$, this chamber has already processed d wafers. If no cleaning operation is scheduled, the robot loads a real wafer in the $(m^1_1 \times d + 1)$th cycle. Instead of loading a real wafer, the robot loads a virtual wafer into this chamber, which is scheduled to be cleaned. In such a case, at the time point $(m^1_1 \times d + 1) \times \Theta$, the robot moves to LLs, and then waits there for $\omega_{1,0}$ time units before unloading a virtual wafer from LLs. Therefore, $t = (m^1_1 \times d + 1) \times \Theta + \mu_1 + \omega_{1,0}$, which means that the robot unloads a virtual wafer from LLs at time point t. Consequently, the chamber with such a virtual wafer can complete its cleaning operation without interruption until it is loaded into a real wafer. Then, the virtual wafer is transferred between chambers step by step with a backward strategy until returning to LLs.

From the above analysis, we can know that, in Situation 1, robot R_1 unloads a virtual wafer from LLs at time point $t = (m^1_1 \times d + 1) \times \Theta + \mu_1 + \omega_{1,0}$ such that the cleaning operation can be performed without interruption in this chamber. Let $PC^i_{f,j}$ denote the chamber f of Step j at C_i that is cleaned. An example shown in Fig. 3 illustrates Situation 1. For a two-CT with a wafer flow pattern (3, 1, 3) in C_1 and (2, 2, 2) in C_2, a condition-based cleaning operation occurs at $PC^1_{3,1}$ after this chamber processes d wafers consecutively. As shown in Fig. 3, in the $(3d + 1)$th cycle, after processing the dth wafer in $PC^1_{3,1}$, the robot loads a virtual wafer from LLs into $PC^1_{3,1}$. Therefore, the cleaning operation can be performed without interruption until the next robot operation at $PC^1_{3,1}$. Then, in the $(3d + 4)$th cycle, a virtual wafer is removed from $PC^1_{3,1}$ and loaded into the next step's chamber, which is loaded/unloaded step by step until returning to LLs in the $(3d + 13)$th cycle.

For Situation 2, the robot needs to unload a virtual wafer from LLs in advance so that a virtual wafer can be loaded into the cleaning chamber on time and the cleaning operation can be completed without interruption. With the known cleaning period d from the controller, the initial cycle of the cleaning operation is also known, the $(d \times m^i_j + 1)$th cycle. With a backward strategy, each wafer can be loaded once. Therefore, if the cleaning operation is performed at $PS_{i,j}$, $i \in N^+_K$, $j \in N^+_{n[i]} \backslash \{b[i]\}$, $j \neq 1$ if $i = 1$, we can know that the virtual wafer in the cleaning chamber is loaded from $PS_{i,(j-1)}$ in the $(d \times m^i_j + 1)$th cycle. Furthermore, this virtual wafer stays in a chamber

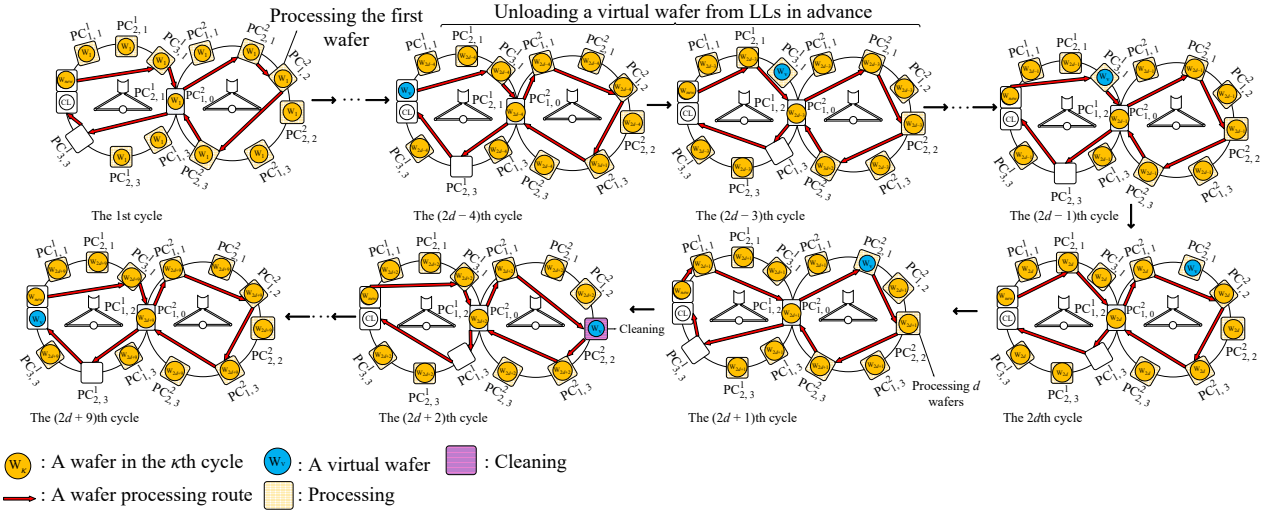

Fig. 4. Example for Situation 2. The condition-based cleaning operation occurs at $PC^2_{2,2}$ after d wafers are processed consecutively.

at $PS_{i,(j-2)}$ in the $((d\times m^i_j + 1) - m^i_{j-1})$th cycle, which implies that this virtual wafer is loaded in the $((d\times m^i_j + 1) - m^i_{j-1})$th cycle, and it stays in the chamber at $PS_{i,(j-3)}$ before being loaded into a chamber at $PS_{i,(j-2)}$. This shows that we can know the time to unload a virtual wafer from LLs by rewinding the position of the virtual wafer. Therefore, the time point t can be calculated by

$$t=\begin{cases}(d\times m^i_j+1-(\sum\limits_{u=0}^{i-1}S^u_{\text{up}}+\sum\limits_{e=1}^{j-1}m^i_e))\times\Theta+\mu_i+\omega_{i,0}, & j<b[i]\\ (d\times m^i_j+1-(S^i_{\text{down}}+\sum\limits_{e=b[i]+1}^{j-1}m^i_e))\times\Theta+\mu_i+\omega_{i,0}, & j>b[i]\\ (d\times m^1_1+1)\times\Theta+\mu_1+\omega_{1,0}, & \text{if } i=1 \text{ and } j=1\end{cases} \tag{17}$$

where $j\in \mathrm{N}^+_{n[i]}\backslash\{b[i]\}$, $i\in \mathrm{N}^+_K$.

If $t<0$, it means that the robot cannot load a virtual wafer into the cleaning chamber when a cleaning operation is performed at $PS_{i,j}$. Note that, if $i=K$, this case is the same as $j<b[i]$.

As shown in Fig. 4, the CT is the same as in Fig. 3. Then, a condition-based cleaning operation occurs at $PC^2_{2,2}$ after d wafers are processed consecutively. Unlike Situation 1, R_1 needs to load a virtual wafer from LLs in advance in the $(2d-4)$th cycle in order to load a virtual wafer into $PC^2_{2,2}$ when a cleaning operation begins at the $(2d+1)$th cycle. The rest of the cycle is similar to Fig. 3 so the explanation is omitted.

During a steady state in a K-CT, the robot waiting time is set by Algorithm 3 such that a feasible schedule is obtained. Besides, the time taken for a robot to load a wafer into a chamber at $PS_{i,j}$, $i\in \mathrm{N}^+_K$, $j\in \mathrm{N}_{n[i]}$, is $(2\lambda_i+3\mu_i+\omega_{i,(j-1)})$ time units. During a transient state, the robot loads a virtual wafer into the cleaning chamber, which takes the same time as that loading a real wafer. After m^i_j cycles, the robot loads a real wafer into this cleaning chamber. Therefore, by loading a virtual wafer, some time is available for cleaning this chamber. Let $\eta_{i,j}$ denote the cleaning time gained by loading a virtual wafer into the cleaning chamber, we have

$$\eta_{i,j}=m^i_j\times\Theta+2\lambda_i+3\mu_i+\omega_{i,(j-1)}, i\in \mathrm{N}^+_K,\ \ j\in \mathrm{N}^+_{n[i]}\backslash\{b[i]\} \tag{18}$$

where m^i_j, λ_i and μ_i are constant in a steady state and a transient state.

By the above analysis, we can know the key to satisfying WRTCs and completing a condition-based cleaning operation is 1) whether a virtual wafer can be properly loaded into a cleaning chamber, which implies $t\geq 0$ should hold; 2) $\eta_{i,j}\geq\rho_{i,j}$ should hold, otherwise a K-CT does not have enough time to complete the chamber cleaning operation that is performed at $PS_{i,j}$, $i\in \mathrm{N}^+_K$, $j\in \mathrm{N}^+_{n[i]}\backslash\{b[i]\}$. Now, it gives rise to another question of whether there is a feasible schedule for a K-CT with a condition-based cleaning operation when $\eta_{i,j}\geq\rho_{i,j}$ does not hold. The answer is positive, to be discussed next.

C. Schedulability for a Condition-Based Cleaning Operation With a Long Cleaning Time

Now, we discuss the case that $\eta_{i,j}<\rho_{i,j}$ holds, which implies that, by loading a virtual wafer into a cleaning chamber, the CT still cannot gain enough cleaning time for such a condition-based cleaning operation that lasts a long time. Consequently, we intend to increase the cycle time by increasing the robot waiting time before unloading a wafer at $PS_{i,1}$ ($\omega_{i,1}$, $i\in \mathrm{N}^+_K$) in each CT. It is obvious that enough time can be obtained for cleaning if the cycle time can be increased before finishing the cleaning operation. However, in such a case, the wafer sojourn time at every step not only increases differently due to different numbers of parallel chambers at each step but also is accumulated gradually. In this circumstance, it is challenging to make WRTCs satisfied. To satisfy WRTCs, we increase the cycle time intermittently by increasing $\omega_{i,1}$ with one increase interval, and we offset the increase in wafer sojourn time by decreasing in $\omega_{i,n[i]}$. In addition, we increase the cycle time intermittently, which means that the frequency of increasing cycle time depends on m^i_2 no matter which step needs a cleaning operation. Let $\lfloor x\rfloor$ be the nearest integer that x is rounded down to. If $x<1$, then $\lfloor x\rfloor=1$.

Let Δ ($\Delta>0$) denote an increment in the cycle time of a K-CT. During a transient state, the wafer sojourn time at $PS_{i,j}$ is denoted by $\tau'_{i,j}$, $i\in \mathrm{N}^+_K$, $j\in \mathrm{N}_{n[i]}$. Furthermore, the robot waiting time before unloading a wafer in the κth cycle during a cleaning operation is denoted by $\omega^\kappa_{i,j}$, where κ is a positive

integer, $i \in \mathrm{N}_K^+$, and $j \in \mathrm{N}_{n[i]}$. For a process-dominant K-CT with WRTCs, we assume that a condition-based cleaning operation occurs in a chamber at PS_{p,j_1}, $p \in \mathrm{N}_K^+$, $j_1 \in \mathrm{N}_{n[i]}^+ \backslash \{b[i]\}$, after this chamber consecutively processes d wafers. Consequently, it means that there is a cleaning requirement for a chamber at Step j_1 in the pth CT (C_p) within a K-CT. Furthermore, we presume that a virtual wafer can be loaded into a cleaning chamber on time.

By the above analysis, with $\eta_{i,j} < \rho_{i,j}$, we increase the cycle time intermittently for $(\lfloor \frac{m_2^p}{2} \rfloor + 1)$ times. Therefore, we set the robots' waiting time by Algorithm 4.

Algorithm 4 Schedule a K-CT for a Cleaning Operation that Lasts a Long Time

Input: $\omega_{i,j}$ $(i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]})$, m_2^p, m_1^p, j_1 (The cleaning step)
Output: $\omega_{i,j}^y$ $(i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]})$, Δ

1: Initialization:
2: **If** $m_2^p + 1 \bmod 2 = 0$ **then** $L \leftarrow 0$ **Else** $L \leftarrow 1$
3: **If** $m_1^p \bmod 2 = 0$ **then** $L_p \leftarrow 0$ **Else** $L_p \leftarrow 1$
4: $\gamma_{p,j_1} \leftarrow 0$
5: **If** $j_1 = 1$ **and** $m_2^p \geq m_1^p$ **then**
6: $\Delta \leftarrow (\rho_{i,j} - \eta_{i,j}) / (\lfloor \frac{m_1^p}{2} \rfloor + L_p)$
7: **Else**
8: $\Delta \leftarrow (\rho_{i,j} - \eta_{i,j}) / (\lfloor \frac{m_2^p}{2} \rfloor + 1)$
9: Set $\omega_{i,j}^y$ as the robot waiting time for a cleaning operation
10: **For** $i \leftarrow 1$ **to** K **do**
/* y denotes the index of the cleaning cycle */
11: $y \leftarrow 1$
12: **while** $y \leq m_2^p + 1$ **do**
13: **If** $y = 1$ **then**
14: **For Each** $j \in \mathrm{N}_{n[i]}^+ \backslash \{1\}$ **do**
15: $\omega_{i,j}^y \leftarrow \omega_{i,j}$
16: $\omega_{i,1}^y \leftarrow \omega_{i,1} + \Delta$
17: **Else**
18: **For Each** $j \in \mathrm{N}_{n[i]}^+ \backslash \{1, n[i]\}$ **do**
19: $\omega_{i,j}^y \leftarrow \omega_{i,j}$
20: $\omega_{i,n[i]}^y \leftarrow \omega_{i,n[i]} - \lfloor \frac{y}{2} \rfloor \Delta$
21: **If** $y = m_2^p + 1$ **and** $j_1 \geq 3$ **and** $i = p$ **then**
22: $\omega_{i,1}^y \leftarrow \omega_{i,1}$
23: $\gamma_{p,j_1} \leftarrow (\lfloor \frac{y}{2} \rfloor + L)\Delta$
24: **Else If** $y \bmod 2 \neq 0$ **then**
25: $\omega_{i,1}^y \leftarrow \omega_{i,1} + (\lfloor \frac{y}{2} \rfloor + 1)\Delta$
26: **Else**
27: $\omega_{i,1}^y \leftarrow \omega_{i,1} + \lfloor \frac{y}{2} \rfloor \Delta$
28: $y \leftarrow y + 1$

For easy presentation, let y' denote the index of the initial cycle of the cleaning operation. By Line 11 in Algorithm 4, we set $y = 1$ at the beginning, which is the same meaning as y'. If $m_1^i \bmod 2 = 0$, then we set $L_i = 0$, $i \in \mathrm{N}_K^+$; otherwise, we set $L_i = 1$, $i \in \mathrm{N}_K^+$, i.e., L_p is set by Line 3 in Algorithm 4, where p means the index of a cluster tool that performs a cleaning operation. Furthermore, L is set by Line 2 in Algorithm 4. The robot waiting time before loading a wafer into a chamber at Step j in C_i is denoted by $\gamma_{i,j}$ so that γ_{p,j_1} is initialized by Line 4 in Algorithm 4.

For a K-CT scheduled by Algorithm 3, we have to ensure that by Algorithm 4, K robots in a K-CT still operate in a paced way so that each pair of CTs share a BC without conflict. Therefore, we have the following conclusion.

Theorem 3: During the cleaning operation, the robot waiting time set by Algorithm 4 still makes (13) and (14) hold, which is the same as that in a steady state.

Proof: 1) For C_i and C_{i+1}, $i \in \mathrm{N}_{K-1}^+$, we presume that the buffer is located at $\mathrm{PS}_{i,2}$, which implies the wafer flow pattern is $(m_1^i, m_{b[i]}^i, m_{b[i]+1}^i, \ldots, m_{n[i]}^i)$, where $b[i] = 2$. By Algorithm 4, when $y = 1$, the increase in $\omega_{i,1}$ means that the event of loading a wafer into $\mathrm{PS}_{i,b[i]}$ is postponed by Δ time units. While in C_{i+1}, the increase in $\omega_{i+1,1}$ means that the event of unloading a wafer from a chamber at $\mathrm{PS}_{i+1,0}$ is also postponed by Δ time units. The unloading wafer at $\mathrm{PS}_{i+1,0}$ is the same wafer at $\mathrm{PS}_{i,b[i]}$, which means that robot R_{i+1} can unload this wafer from $\mathrm{PS}_{i+1,0} = \mathrm{PS}_{i,b[i]}$ in C_i because R_i performs its action towards $\mathrm{PS}_{i,b[i]}$ with the same postponement. Therefore, during the y'th cycle ($y = 1$), the increases in $\omega_{i,1}$ and $\omega_{i+1,1}$ do not break the synchronization between C_i and C_{i+1}, which implies that (13) and (14) still hold. Next, during the yth ($2 \leq y \leq m_2^p$) cycle in C_i, by Line 20 in Algorithm 4, the decrease in $\omega_{i,n[i]}$ means that the event of unloading a wafer in $\mathrm{PS}_{i,b[i]}$ is performed $\lfloor \frac{y}{2} \rfloor \Delta$ earlier than usual since the other robot waiting time remains unchanged except for the robot's waiting time at $\mathrm{PS}_{i,1}$. While in C_{i+1}, the decrease in $\omega_{i+1,n[i+1]}$ means that loading a wafer at $\mathrm{PS}_{i+1,0}$ is also performed $\lfloor \frac{y}{2} \rfloor \Delta$ earlier than usual. The unloading wafer at $\mathrm{PS}_{b[i]}$ is the same as that at $\mathrm{PS}_{i+1,0}$, which implies that robot R_i can unload this wafer from the chamber at $\mathrm{PS}_{i+1,0}$ because the loading operation is also performed $\lfloor \frac{y}{2} \rfloor \Delta$ earlier in C_{i+1}. The increases in $\omega_{i,1}$ and $\omega_{i+1,1}$ are similar to the increase in the y'th cycle ($y = 1$) such that the neighboring robots operate in a paced way. When $y = m_2^p + 1$, the coordination between C_i and C_{i+1}, $i \in \mathrm{N}_{K-1}^+ \backslash \{p\}$ is still valid because the setting in $\omega_{i,n[i]}$ and $\omega_{i+1,n[i+1]}$ is similar to the yth cycle, which makes the same postponement such that (13) and (14) still hold. As for C_p and C_{p+1}, if $j_1 \geq 3$, the robot waiting time before unloading a wafer at $\mathrm{PS}_{p,1}$ remains unchanged by Line 22 in Algorithm 4. However, the loading action of a wafer at $\mathrm{PS}_{p,b[p]}$ is still postponed by $(\lfloor \frac{y}{2} \rfloor + L)\Delta$ time units. The loading operation at PS_{p,j_1} is postponed by $(\lfloor \frac{y}{2} \rfloor + L)\Delta$ time units under the setting of Line 23 in Algorithm 4, and it is performed before the loading operation at $\mathrm{PS}_{p,b[p]}$. Since unloading a wafer at $\mathrm{PS}_{(p+1),1}$ is postponed by $(\lfloor \frac{y}{2} \rfloor + 1)\Delta$ or $\lfloor \frac{y}{2} \rfloor \Delta$ time units that is the same as $(\lfloor \frac{y}{2} \rfloor + L)\Delta$, robot R_{p+1} also postpones its actions. Then, robot R_{p+1} still can unload the wafer from $\mathrm{PS}_{p+1,0}$ ($\mathrm{PS}_{p,b[p]}$) due to (13) and (14) still holding. If $j_1 < 3$, the effect of the increases in $\omega_{i,1}$ and $\omega_{i+1,1}$ is similar to the yth cycle, which makes (13) and (14) hold.

2) For C_i and C_{i+1}, $i \in \mathrm{N}_{K-1}^+$, we presume that the buffer location is at Step j_2, $j_2 \in \mathrm{N}_{n[i]}^+ \backslash \{1, 2, n[i]\}$, which implies that the wafer flow pattern is $(m_1^i, m_2^i, \ldots, m_{b[i]}^i, m_{j_2+1}^i, \ldots, m_{n[i]}^i)$. By Algorithm 4, when $y = 1$, for C_{i+1}, the increase in $\omega_{i+1,1}$ means that the event of unloading a wafer from a chamber at $\mathrm{PS}_{(i+1),0}$ is postponed by Δ time units. While in C_i, the increase in $\omega_{i,1}$ does not affect the wafer in a chamber at $\mathrm{PS}_{i,b[i]}$

because the wafer has already been loaded into a chamber at $PS_{i,b[i]}$ before $\omega_{i,1}$ gets increased. Therefore, the coordination is still valid between C_i and C_{i+1} in the first cycle of a cleaning operation.

Next, during the yth ($2 \le y \le m_2^p$) cycle, in C_i and C_{i+1}, the decreases in $\omega_{i,n[i]}$ and $\omega_{i+1,n[i+1]}$ make (13) and (14) hold between C_i and C_{i+1} because robot R_i can unload the wafer from the chamber at $PS_{(i+1),0}$. The increases in $\omega_{i,1}$ and $\omega_{i+1,1}$ also satisfy (13) and (14), which is the same as in the y'th cycle. When $y = m_2^p + 1$, similarly, the decreases in $\omega_{i,n[i]}$ and $\omega_{i+1,n[i+1]}$ make the C_i and C_{i+1} still coordinate well with each other. For C_i and C_{i+1}, $i \in \mathrm{N}_{K-1}^+ \backslash \{p\}$, the coordination is still valid because the setting in $\omega_{i,1}$ and $\omega_{i+1,1}$ is similar to the setting in the previous cycle, which means that the effect of the postponement is the same so that (13) and (14) still hold in such CTs. As for C_p and C_{p+1}, if $j_1 < j_2$, the effect of the setting is the same as the effect of the increase in $\omega_{i,1}$ and $\omega_{i+1,1}$ in the yth ($2 \le y \le m_2^p + 1$) cycle, which means that (13) and (14) are met. If $j_1 > j_2$, the effect of the setting is the same as the influence of the increase in $\omega_{i,1}$ and $\omega_{i+1,1}$ in the yth ($2 \le y \le m_2^p + 1$) cycle mentioned in i), which means that the setting satisfies (13) and (14). ■

The above analysis shows that a K-CT satisfies (13) and (14) between two adjacent CTs if they are already met during a steady state after the robots' waiting time is set by Algorithm 4.

Note that, with L_i, Δ is set by Lines 5 to 8 in Algorithm 4. After the cycle time is intermittently increased for $(\lfloor \frac{m_2^p}{2} \rfloor + 1)$ times, we presume that the wafer sojourn time at $PS_{i,3}$ to $PS_{i,n[i]}$ remains unchanged, which can be demonstrated by the following Lemma 1. Therefore, if a condition-based cleaning operation with a long cleaning time is performed at one step of Steps $PS_{i,3}$ to $PS_{i,n[i]}$, we increase the robot waiting time before loading a wafer at such a step $(\gamma_{i,j})$ by $(\lfloor \frac{m_2^p}{2} \rfloor + 1)\Delta$ time units in Algorithm 4 to obtain extra time for cleaning. With such a cleaning operation performed at $PS_{i,2}$, we can increase the robot waiting time before unloading a wafer at $PS_{i,1}$ $(\omega_{i,1})$ by $(\lfloor \frac{m_2^p}{2} \rfloor + 1)\Delta$ time units to attain time for cleaning. Consequently, when a cleaning operation is performed at one step of Steps $PS_{i,2}$ to $PS_{i,n[i]}$, $\Delta = (\rho_{i,j_1} - \eta_{i,j_1}) / (\lfloor \frac{m_2^p}{2} \rfloor + 1)$ is set by Line 8 in Algorithm 4. The wafer sojourn time at $PS_{i,1}$ is increased during the cleaning operation by Algorithm 4. Thus, if the cleaning operation is performed at $PS_{i,1}$, we need to calculate the increment at $PS_{i,1}$ so that Δ is set. According to the condition between m_2^p and m_1^p, $\Delta = (\rho_{i,j_1} - \eta_{i,j_1}) / (\lfloor \frac{m_2^p}{2} \rfloor + 1)$ or $(\rho_{i,j_1} - \eta_{i,j_1}) / (\lfloor \frac{m_1^p}{2} \rfloor + L_p)$ is set by Lines 5 to 8 in Algorithm 4 to obtain extra time for the performed cleaning operation at $PS_{i,1}$. When $m_2^p \ge m_1^i$, we set $x_1^i = (\lfloor \frac{m_1^i}{2} \rfloor + L_i)$, $i \in \mathrm{N}_K^+$; otherwise, we set $x_0^i = (\lfloor \frac{m_2^p}{2} \rfloor + 1)$, $i \in \mathrm{N}_{K^+}$.

Let $\|D_{o_1,o_2}\|$ denote the number of cycles from the o_1th cycle to the o_2th cycle, o_1, $o_2 \in \mathrm{N}_{m_2^p+1}^+$, e.g., $\|D_{2,4}\| = 3$. Furthermore, if y mod 2 = 0, then we set $I_y = 0$; otherwise, we set $I_y = 1$. During the first cycle of the cleaning operation ($y = 1$) to $(m_2^p + 1)$th cycle, we have the following conclusion by Algorithm 4.

Lemma 1: By Line 20 in Algorithm 4, during a cleaning operation, the wafer sojourn time at $PS_{i,3}$ to $PS_{i,n[i]}$, $i \in \mathrm{N}_{\mathrm{K}}^+$, remains unchanged.

Proof: In the y_1th ($1 < y_1 \le m_2^p$) cycle, a wafer is loaded into a chamber at $PS_{i,m_{w_1}^i}$, where $3 \le w_1 \le n[i]$. After $m_{w_1}^i$ cycles, or in the $(m_{w_1}^i + y_1)$th ($m_{w_1}^i + y_1 \le m_2^p + 1$) cycle, a robot unloads this wafer from a chamber at $PS_{i,m_{w_1}^i}$. Before such an unloading operation, the robot waiting time at $PS_{i,n[i]}$ is decreased by $\lfloor \frac{m_{w_1}^i + y_1}{2} \rfloor \Delta$ time units by Line 20 in Algorithm 4. $\lfloor \frac{m_{w_1}^i + y_1}{2} \rfloor \Delta$ is also the total increment from the y'th cycle to the $(m_{w_1}^i + (y_1 - 1))$th cycle, which can be divided into two parts:

1) The increment placed between the y'th cycle and y_1th cycle: $(\lfloor \frac{y_1}{2} \rfloor + I_{y_1})\Delta$;

2) The increment placed between the $(y_1 + 1)$th cycle and the $(m_{w_1}^i + (y_1 - 1))$th cycle: $(\lfloor \| D_{y_1+1, m_{w_1}^i + (y_1-1)} \| / 2 \rfloor + I_{y_1+1})\Delta = (\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta$. Thus,

$$\lfloor \frac{m_{w_1}^i + y_1}{2} \rfloor \Delta = (\lfloor \frac{y_1}{2} \rfloor + I_{y_1})\Delta + (\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta.$$

Because a wafer is loaded into a chamber in the y_1th cycle and unloaded in the $(m_{w_1}^i + y_1)$th cycle where $m_{w_1}^i + y_1 \le m_2^p + 1$, the total increment in such a wafer's sojourn time can be also divided into two parts:

1) The increment between the $(y_1 + 1)$th cycle and the $(m_{w_1}^i + (y_1 - 1))$th cycle is the same as shown above, which is $(\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta$;

2) The increment in the y_1th cycle, which is set by Lines 24 to 27 in Algorithm 4 during $y = y_1$.

Therefore, the total increment in such a wafer's sojourn time is $(\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta + \lfloor \frac{y_1}{2} \rfloor \Delta$ or $(\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta + (\lfloor \frac{y_1}{2} \rfloor + 1)\Delta$. Thus, if y_1 mod 2 = 0, then

$$\tau'_{i,w_1} = \tau_{i,w_1} + (\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta + \lfloor \frac{y_1}{2} \rfloor \Delta.$$

Or, if y_1 mod $2 \ne 0$, then

$$\tau'_{i,w_1} = \tau_{i,w_1} + (\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta + (\lfloor \frac{y_1}{2} \rfloor + 1)\Delta.$$

The total increment in such a wafer's sojourn time and the total increment from the y'th cycle to the $(m_{w_1}^i + (y_1 - 1))$th cycle is the same as shown in the above analysis. By Line 20 in Algorithm 4, the robot waiting time before unloading a wafer at $PS_{i,n[i]}$ is decreased by $\lfloor \frac{m_{w_1}^i + y_1}{2} \rfloor \Delta$ that is the total increment from the y'th cycle to the $(m_{w_1}^i + (y_1 - 1))$th cycle, and the unloading operation at $PS_{i,m_{w_1}^i}$ is performed after the unloading operation at $PS_{i,n[i]}$. Thus, if y_1 mod 2 = 0, then $I_{y_1} = 0$, and

$$\begin{aligned}\tau'_{i,w_1} &= \tau_{i,w_1} + (\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta + \lfloor \frac{y_1}{2} \rfloor \Delta - \lfloor \frac{m_{w_1}^i + y_1}{2} \rfloor \Delta \\ &= \tau_{i,w_1} + (\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta + \lfloor \frac{y_1}{2} \rfloor \Delta \\ &\quad - ((\lfloor \frac{y_1}{2} \rfloor + I_{y_1})\Delta + (\lfloor \frac{m_{w_1}^i - 1}{2} \rfloor + I_{y_1+1})\Delta) = \tau_{i,w_1}.\end{aligned}$$

Or, if $y_1 \bmod 2 \neq 0$, then $I_{y_1} = 1$, and

$$\tau'_{i,w_1} = \tau_{i,w_1} + (\lfloor \frac{m^i_{w_1} - 1}{2} \rfloor + I_{y_1+1})\Delta + (\lfloor \frac{y_1}{2} \rfloor + 1)\Delta - \lfloor \frac{m^i_{w_1} + y_1}{2} \rfloor \Delta$$

$$= \tau_{i,w_1} + (\lfloor \frac{m^i_{w_1} - 1}{2} \rfloor + I_{y_1+1})\Delta + (\lfloor \frac{y_1}{2} \rfloor + 1)\Delta$$

$$- ((\lfloor \frac{y_1}{2} \rfloor + I_{y_1})\Delta + (\lfloor \frac{m^i_{w_1} - 1}{2} \rfloor + I_{y_1+1})\Delta) = \tau_{i,w_1}.$$

Therefore, during a cleaning operation, the wafer sojourn time at $PS_{i,3}$ to $PS_{i,n[i]}$, $i \in N_K^+$, remains unchanged with the setting in Algorithm 4. ■

The above analysis shows that the wafer sojourn time of PS_{i,w_1} $i \in N_K^+$, $3 \leq w_1 \leq n[i]$, remains unchanged by Algorithm 4. Therefore, if the cleaning operation occurs at PS_{p,w_1}, we increase the robot waiting time before loading a wafer into a chamber at PS_{p,w_1} (γ_{p,w_1}) to get enough time for cleaning the chamber at PS_{p,w_1}.

Furthermore, by Algorithm 4, the following conditions should hold to make a K-CT satisfy WRTCs:

$$\omega_{i,n[i]} \geq (\lfloor \frac{m_2^p}{2} \rfloor + 1)\Delta, \ i \in N_K^+ \tag{19}$$

$$\alpha_{i,1} + \delta_{i,1} - \tau_{i,1} \geq x_r^i \Delta, \ r \in \{0,1\}, \ i \in N_K^+ \tag{20}$$

$$\tau_{i_1,2} - \alpha_{i_1,2} \geq \Delta, \text{ if } m_2^p > m_2^{i_1}, \ i_1 \in N_K^+ \setminus \{p\} \tag{21}$$

and

$$\tau_{p,2} - \alpha_{p,2} \geq \Delta, \text{ if } j_1 \neq 2. \tag{22}$$

Theorem 4: During a cleaning operation, if (19)–(22) hold, then we can increase the cycle time for a K-CT by Algorithm 4, which ensures that WRTCs of a K-CT are still satisfied.

Proof: During the first cycle of the cleaning operation ($y = 1$), for C_i, $i \in N_K^+$, when robot R_1 performs the waiting action at $PS_{i,1}$ ($\omega_{i,1}$), each processed wafer has been loaded into a chamber at the next step except the wafer at $PS_{i,1}$. These processed wafers satisfy WRTCs because an increase in $\omega_{i,1}$ is given after the robot unloads these wafers, which implies that the sojourn time of these wafers remains unchanged. The increase in $\omega_{i,1}$ prolongs the wafer sojourn time of the WIP wafers that are unloaded in the next cycle. Therefore, after the y'th cycle, we offset the increase in $\omega_{i,1}$ that is given in the previous cycles by decreasing the robot waiting time before unloading a wafer from a chamber at $PS_{i,n[i]}$ ($\omega_{i,n[i]}$). By Lemma 1, until the last cycle ($y = m_2^p + 1$) of a cleaning operation, the wafer sojourn time at $PS_{i,3}$ to $PS_{i,n[i]}$ remains unchanged. If (19) holds, it means that we can offset such increases by decreasing $\omega_{i,n[i]}$ during the cleaning operation. Thus, the wafer sojourn time at these steps remains unchanged so that WRTCs of these steps are satisfied.

By Algorithm 4, the cycle time is intermittently increased during a cleaning operation, while $\omega_{i,0}^y$ remains unchanged. If $m_2^p \geq m_1^i$, in the m_1^ith cycle, the total increment in cycle time at $PS_{i,1}$ is $\lfloor \| D_{y',m_1^i} \| /2 \rfloor + I_{m_1^i} = x_1^i = (\lfloor \frac{m_1^i}{2} \rfloor + L_i)\Delta$; otherwise, such an increment is $x_0^i = (\lfloor \frac{m_2^p}{2} \rfloor + 1)$. Thus, according to (8), the wafer sojourn time at $PS_{i,1}$ is $\tau'_{i,1} = \Theta \times m_1^i + x_r^i \Delta - (4\lambda_i + 3\mu_i + \omega_{i,0}^{m_1^i}) = \tau_{i,1} + x_r^i \Delta$. If (20) holds, it means that $\tau'_{i,1} = \tau_{i,1} + x_r^i \Delta \leq \alpha_{i,1} + \delta_{i,1}$, namely, WRTCs at $PS_{i,1}$ are satisfied.

For the wafer sojourn time at $PS_{i,2}$, we have the following situations.

1) For $m_2^p < m_2^{i_1}$, $i_1 \in N_K^+ \setminus \{p\}$, from the y'th to the $(m_2^p + 1)$th cycle, the increment in wafer sojourn time at $PS_{i_1,2}$ is the same as that at $PS_{i_1,3}$ to $PS_{i_1,n[i]}$. Thus, if (19) holds, the decrease in $\omega_{i_1,n[i]}$ can offset the increase in $\omega_{i_1,1}$ that is given in the previous cycles, which means that WRTCs at $PS_{i_1,2}$ are satisfied.

2) For $m_2^p \geq m_2^{i_1}$, $i_1 \in N_K^+ \setminus \{p\}$, in the $(m_2^{i_1} + 1)$th cycle, by Algorithm 4, the total increment in cycle time at $PS_{i_1,2}$ is $\lfloor \| D_{y',m_2^{i_1}+1} \| /2 \rfloor \Delta = (\lfloor \frac{m_2^{i_1}+1}{2} \rfloor)\Delta$, while $\omega_{i_1,1}^{m_2^{i_1}+1}$ is increased by Lines 24–27. Therefore, following (8), the wafer sojourn time at $PS_{i_1,2}$ is

$$\tau'_{i_1,2} = \Theta \times m_2^{i_1} + (\lfloor \frac{m_2^{i_1}+1}{2} \rfloor)\Delta - (4\lambda_i + 3\mu_i + \omega_{i_1,1}^{m_2^{i_1}+1})$$

$$= \Theta \times m_2^{i_1} + (\lfloor \frac{m_2^{i_1}+1}{2} \rfloor)\Delta - (4\lambda_i + 3\mu_i$$

$$+ (\lfloor \frac{m_2^{i_1}+1}{2} \rfloor + 1)\Delta + \omega_{i_1,1}) = \tau_{i_1,2} - \Delta.$$

Or

$$\tau'_{i_1,2} = \Theta \times m_2^{i_1} + (\lfloor \frac{m_2^{i_1}+1}{2} \rfloor)\Delta - (4\lambda_i + 3\mu_i$$

$$+ (\lfloor \frac{m_2^{i_1}+1}{2} \rfloor)\Delta + \omega_{i_1,1}) = \tau_{i_1,2}.$$

If (21) holds and $\tau'_{i_1,2} = \tau_{i_1,2} - \Delta$, it means that

$$\tau'_{i_1,2} = \tau_{i_1,2} - \Delta \geq \alpha_{i_1,2},$$

namely, WRTCs at $PS_{i_1,2}$ are satisfied.

3) For C_p, if $j_1 = 2$, it means that the cleaning operation occurs at $PS_{p,2}$ which is loaded with a virtual wafer that is not imposed on WRTCs. Therefore, when $j_1 = 2$, WRTCs at $PS_{p,2}$ always are satisfied; otherwise, the effect of the decrease of $\omega_{i,n[i]}$ is the same as that mentioned in 2), which implies that the wafer sojourn time at $PS_{p,2}$ is also decreased by Δ time units. If (22) holds, WRTCs at $PS_{p,2}$ are satisfied. ■

For $m_2^p < m_2^{i_1}$, $i_1 \in N_K^+ \setminus \{p\}$, when $j_1 = 2$, (19) and (20) should hold for satisfying WRTCs of each processing step in a K-CT during a cleaning operation. When $j_1 \neq 2$, (19), (20) and (22) should hold.

For $m_2^p \geq m_2^{i_1}$, $i_1 \in N_K^+ \setminus \{p\}$, when $j_1 = 2$, (19)–(21) should hold for satisfying WRTCs for each processing step in a K-CT during a cleaning operation. When $j_1 \neq 2$, (19)–(22) should hold.

Theorem 5: A feasible schedule can be found by Algorithm 4 during the cleaning operation of a single-arm MCT with parallel chambers and WRTCs, if and only if (19)–(22) are satisfied.

Proof (Necessity): According to the proof of Theorem 4, if a feasible schedule can be found by Algorithm 4, the wafer sojourn time of each step must satisfy WRTCs, i.e., Condi-

tions (19)−(22) must hold. Therefore, (19)−(22) are necessary.

Proof (Sufficiency): By the proof of Theorem 4, if $m_2^p < m_2^{i_1}$, $i_1 \in \mathrm{N}_K^+ \backslash \{p\}$ and $j_1 = 2$, then $\omega_{i,n[i]} \geq (\lfloor \frac{m_2^p}{2} \rfloor + 1)\Delta$, $i \in \mathrm{N}_K^+$, guarantees that the wafer sojourn time at $\mathrm{PS}_{i,n[i]}$ to $\mathrm{PS}_{i,2}$ satisfies WRTCs because their increments except for $\mathrm{PS}_{i,1}$ are the same as that at $\mathrm{PS}_{i,n[i]}$. Thus, $\alpha_{i,1} + \delta_{i,1} - \tau_{i,1} \geq x_r^i \Delta$, $i \in \mathrm{N}_K^+$, pledges that WRTCs at $\mathrm{PS}_{i,1}$ are satisfied. If $m_2^p \geq m_2^{i_1}$, $i_1 \in \mathrm{N}_K^+ \setminus \{p\}$, and $j_1 \neq 2$, $\tau_{i,2} - \alpha_{i,2} \geq \Delta$, $i \in \mathrm{N}_K^+$ hold, they ensure that the WRTCs at $\mathrm{PS}_{i,2}$ can be satisfied during the cleaning operation. Consequently, (19)−(22) are sufficient. ■

In Algorithm 4, the number of iterations in the For-loop, i.e., Lines 14−15 and 1−19, is no more than $n[i]$. In addition, by Line 12, the number of iterations in such a While-loop is no more than $(m_2^p + 1)$, which is a constant. Thus, combined with the For-loop in Line 10, the computational complexity of Algorithm 4 is polynomial.

After the cleaning operation, we need to set the robot waiting time properly to make the K-CT return to a steady state. To do so, we propose an algorithm in the next section.

D. Algorithms for Switching a Transient State to a Steady State

Algorithm 5 Schedule a K-CT After a Condition-Based Cleaning Operation

Input: $\omega_{i,j}$ $(i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]})$, Δ
Output: $\zeta_{i,j}^{\iota_i}$, Q_1
1: Set $\zeta_{i,j}$ as the robot waiting time for switching to a steady state
2: **For** $i \leftarrow 1$ **to** K **do**
3: $B_1 \leftarrow 0$, $i_2 \leftarrow 1$
4: **ForEach** $j \in \mathrm{N}_{n[i]}$ **do** $\upsilon_{i,j} \leftarrow 0$, $\zeta_{i,j}^{\iota_i} \leftarrow 0$
5: **For** $s \leftarrow 1$ **to** $m_{n[i]}^i$ **do**
6: $B \leftarrow 0$
7: **If** $s = m_{j_1}^p$ **and** $i = p$ **and** $j_1 \geq 3$ **then**
8: $B_1 \leftarrow 1$
9: $\iota_i \leftarrow m_2^p + 1 + s$
10: $\sigma_{i,j}^{\iota_i} \leftarrow 0$, $\iota \leftarrow 1$
11: Call Algorithm 6
12: $\iota' \leftarrow \max\{\iota_{i_1} \mid 1 \leq i_1 \leq i\}$
13: **For** $\kappa \leftarrow m_2^p + 2$ **to** ι' **do**
14: $Q_1 \leftarrow$ Algorithm 7
15: **End**

After the cleaning operation in the $(m_2^p + 1)$th cycle is finished, the K-CT needs to make a transition from a transient state to a steady state. In this section, we propose Algorithm 5 to achieve this transition by properly adjusting the robot waiting time in each CT. During such a transition, the robot waiting time in the κth cycle is denoted by $\zeta_{i,j}^{\kappa}$, $i \in \mathrm{N}_K^+$, $j \in \mathrm{N}_{n[i]}$, $\kappa \in \mathrm{N}^+$. Let $\sigma_{i,j}^{\kappa}$ denote the adjustment of the robot waiting time at $\mathrm{PS}_{i,j}$ in the κth cycle, $i \in \mathrm{N}_K^+$, $j \in \mathrm{N}_{n[i]}^+$, $\kappa \in \mathrm{N}^+$, and $\zeta_{i,j}^{\kappa} = \omega_{i,j} + \sigma_{i,j}^{\kappa}$. Furthermore, υ_{i,w_1} is the total adjustment of the robot waiting time from $\mathrm{PS}_{i,1}$ to PS_{i,w_1-2}, $i \in \mathrm{N}_K^+$, $3 \leq w_1 \leq n[i]$. In particular, $\upsilon_{i,1} = \upsilon_{i,2} = 0$. Firstly, Lines 3−10 in Algorithm 5 initialize the robot waiting time. Then, Algorithm 6 sets the robot waiting time for each CT to switch from a transient state to a steady state. Since the difference of each parallel chamber number is small, we set the robot waiting time in every $m_{n[i]}^i$ cycles for each step until a CT switches to a steady state by Algorithm 6. ι' denotes the biggest index of cycles after such a transition. Next, Line 12 in Algorithm 5 sets ι', and we can check the coordination of a buffer chamber between two adjacent CTs from the $(m_2^p + 2)$th cycle to the ι'th cycle by Algorithm 7.

Algorithm 6 Set the Robot Waiting Time for a K-CT Switches to a Steady State

Input: $\omega_{i,j}$, $\upsilon_{i,j}$, $\sigma_{i,j}^{\iota_i}$ $(i \in \mathrm{N}_K^+, j \in \mathrm{N}_{n[i]})$, Δ, B, B_1
Output: $\zeta_{i,j}^{\iota_i}$, $\sigma_{i,j}^{\iota_i}$, ι_i
1: **Do**
2: **If** $n[i] = 2$ **then** $z_1 \leftarrow \Delta$, $B \leftarrow B + 1$ **Else** $z_1 \leftarrow (\lfloor \frac{m_2^p}{2} \rfloor + 1)\Delta$
3: **If** $\iota = 1$ **then**
4: **If** $B_1 = 1$ **then** goto Line 7
5: $\zeta_{i,n[i]}^{\iota_i} \leftarrow \omega_{i,n[i]} - z_1$, $\sigma_{i,n[i]}^{\iota_i} \leftarrow -z_1$
6: **Else**
7: $\zeta_{i,n[i]}^{\iota_i} \leftarrow \omega_{i,n[i]} - \upsilon_{i,n[i]}$, $\sigma_{i,n[i]}^{\iota_i} \leftarrow -\upsilon_{i,n[i]}$
8: **For** $s_1 \leftarrow n[i] - 1$ **down to** 1 **and** $s_1 \neq b[i]$ **do**
9: $e_2 \leftarrow \iota_i - m_{s_1}^i$
10: **If** $\iota = 1$ **and** $s_1 > 2$ **then**
11: **If** $B_1 = 1$ **then**
12: $\zeta_{i,s_1}^{\iota_i} \leftarrow \omega_{i,s_1} + \sigma_{i,n[i]}^{\iota_i}$, $\sigma_{i,s_1}^{\iota_i} \leftarrow \sigma_{i,n[i]}^{\iota_i}$
13: **Else**
14: $\zeta_{i,s_1}^{\iota_i} \leftarrow \omega_{i,s_1}$, $\sigma_{i,s_1}^{\iota_i} \leftarrow 0$
15: **Else**
16: **If** $e_2 < m_2^p + 1$ **then**
17: **If** e_2 **mod** $2 = 0$ **then** $e_2 \leftarrow \lfloor \frac{e_2}{2} \rfloor$
18: **Else** $e_2 \leftarrow \lfloor \frac{e_2}{2} \rfloor + 1$
19: $part \leftarrow z_1 - e_2$
20: $\zeta_{i,s_1}^{\iota_i} \leftarrow \omega_{i,s_1} + |part \times \Delta + \sum_{e=s_1+1}^{n[i]} \sigma_{i,e}^{\iota_i} - \sigma_{i,n[i]}^{\iota_i}|$
21: $\sigma_{i,s_1}^{\iota_i} \leftarrow |part \times \Delta + \sum_{e=s_1+1}^{n[i]-1} \sigma_{i,e}^{\iota_i} - \sigma_{i,n[i]}^{\iota_i}|$
22: **Else if** $e_2 \geq m_2^p + 1$ **and** $s_1 > 2$ **then**
23: $\zeta_{i,s_1}^{\iota_i} \leftarrow \omega_{i,s_1} + |\upsilon_{i,s_1} + \sum_{e=s_1+1}^{n[i]-1} \sigma_{i,e}^{\iota_i} - \sigma_{i,n[i]}^{\iota_i}|$
24: $\sigma_{i,s_1}^{\iota_i} \leftarrow |\upsilon_{i,s_1} + \sum_{e=s_1+1}^{n[i]-1} \sigma_{i,e}^{\iota_i} - \sigma_{i,n[i]}^{\iota_i}|$
25: **Else**
26: $\zeta_{i,s_1}^{\iota_i} \leftarrow \omega_{i,s_1} + (\sigma_{i,n[i]}^{\iota_i} - \sum_{e=s_1+1}^{n[i]-1} \sigma_{i,e}^{\iota_i} - \sigma_{i,b[i]+1}^{\iota_i})$
27: $\sigma_{i,s_1}^{\iota_i} \leftarrow \sigma_{i,n[i]}^{\iota_i} - \sum_{e=s_1+1}^{n[i]-1} \sigma_{i,e}^{\iota_i} - \sigma_{i,b[i]+1}^{\iota_i}$
28: **if** $i \neq K$ **then**
29: **If** $b[i] = 2$ **or** $\sigma_{i,b[i]-1}^{\iota_i} = 0$ **then**
30: $\zeta_{i,b[i]}^{\iota_i} \leftarrow \omega_{i,b[i]} + (\sigma_{i,n[i]}^{\iota_i} - \sum_{e \in \mathrm{N}_{n[i]-1 \backslash \{b[i]\}}^+} \sigma_{i,e}^{\iota_i})$
31: $\sigma_{i,b[i]}^{\iota_i} \leftarrow \sigma_{i,n[i]}^{\iota_i} - \sum_{e \in \mathrm{N}_{n[i]-1 \backslash \{b[i]\}}^+} \sigma_{i,e}^{\iota_i}$
32: **Else**
33: $\zeta_{i,b[i]}^{\iota_i} \leftarrow \omega_{i,b[i]} + \sigma_{i,b[i]-1}^{\iota_i}$, $\sigma_{i,b[i]}^{\iota_i} \leftarrow \sigma_{i,b[i]-1}^{\iota_i}$
34: $\zeta_{i,b[i]-1}^{\iota_i} \leftarrow \omega_{i,b[i]-1}$, $\sigma_{i,b[i]-1}^{\iota_i} \leftarrow 0$
35: $\zeta_{i,1}^{\iota_i} \leftarrow \zeta_{i,1}^{\iota_i} + (\sigma_{i,n[i]}^{\iota_i} - \sum_{e \in \mathrm{N}_{n[i]-1}^+} \sigma_{i,e}^{\iota_i})$
36: $\sigma_{i,1}^{\iota_i} \leftarrow \sigma_{i,n[i]}^{\iota_i} - \sum_{e \in \mathrm{N}_{n[i]-1}^+} \sigma_{i,e}^{\iota_i}$
37: **For** $s_2 \leftarrow n[i]$ **down to** 3 **do**
38: $\upsilon_{i,s_2} \leftarrow \sum_{e_1=1}^{s_2-2} \sigma_{i,e_1}^{\iota_i}$
39: **If** $\upsilon_{i,s_2} = 0$ **then** $B \leftarrow B + 1$
40: $\iota \leftarrow \iota + 1$, $\iota_i \leftarrow \iota_i + m_{n[i]}^i$
41: **While** $B < n[i] - 2$;

Algorithm 7 Check the coordination between C_{i-1} and C_i, $i \in \mathrm{N}_K^+$

Input: $\sigma^{\kappa}_{i,j}$
Output: Q_1
1: **if** $i \geq 2$ **then**
2: **while** $i_2 \leq i$ **do**
3: $\Phi_1 \leftarrow \sum_{e_1=1}^{b[i-1]-2} \sigma^{\kappa-1}_{i-1,e_2} + \sum_{e_2=b[i-1]}^{n[i-1]} \sigma^{\kappa}_{i-1,e_2}$, $\Phi_2 \leftarrow \sigma^{\kappa}_{i,n[i]}$
4: $\Phi_3 \leftarrow \sum_{e_3=1}^{n[i]-1} \sigma^{\kappa}_{i,e_3}$, $\Phi_4 \leftarrow \sigma^{\kappa}_{i-1,b[i-1]-1}$
5: **If** $\Phi_1 \geq \Phi_2$ **and** $\Phi_3 \geq \Phi_4$ **then**
/* C_{i-1} and C_i can switch to a steady state */
6: $Q_1 \leftarrow 0$
7: **Else**
/* C_{i-1} and C_i can not switch to a steady state */
8: $Q_1 \leftarrow i$
9: $i_2 \leftarrow i_2 + 1$

The key to realizing such a transition for a K-CT is how to adjust the robot waiting time without violating WRTCs and keep the robot waiting time of $\mathrm{PS}_{i,1}$ to $\mathrm{PS}_{i,(n[i]-2)}$ unchanged, $n[i] > 2$. In a cycle, the robot waiting time at $\mathrm{PS}_{i,j}$, $j \in \mathrm{N}^+_{n[i]-2}$, affects the wafer sojourn time at $\mathrm{PS}_{i,(j+2)}$ instead of the wafer sojourn time at $\mathrm{PS}_{i,(j+1)}$ since the chamber at $\mathrm{PS}_{i,(j+1)}$ remains empty with a backward strategy. Therefore, from Lines 39 to 40, we calculate the increase in $\mathrm{PS}_{i,(n[i])}$ to $\mathrm{PS}_{i,3}$ for the restoration of the next cycle. By Algorithm 4, from the $(m_2^p + 2)$th cycle to the $(m_2^p + 1 + m^i_{n[i]})$th cycle, the total increase in the wafer sojourn time of $\mathrm{PS}_{i,(n[i])}$ is the same, which is $(\lfloor \frac{m_2^p}{2} \rfloor + 1)\Delta$. When $n[i] = 2$, the total increment in the wafer sojourn time of $\mathrm{PS}_{i,(n[i])}$ is Δ. We offset such increases with decreasing $\omega_{i,n[i]}$ by Lines 5−7. The decrease in $\omega_{i,n[i]}$ means that the robot performs the next action earlier after the robot waiting event at $\mathrm{PS}_{i,(n[i])}$, which implies that the increase in wafer sojourn time in the rest of the steps also gets offset. Therefore, by Lines 8−27, we calculate the total increase in the wafer sojourn time of $\mathrm{PS}_{i,1}$ to $\mathrm{PS}_{i,n[i]-1}$, except $\mathrm{PS}_{i,b[i]}$, and then offset the increase by setting the robot waiting time of these steps to satisfy WRTCs. The robot waiting time at $\mathrm{PS}_{i,b[i]}$, $i \neq K$, is set by Lines 28−36. If $b[i] = 2$, the increment in $\omega_{i,b[i]}$ is the rest of the total waiting time. If $b[i] > 2$, the increment in $\omega_{i,b[i]}$ is the same as that in $\omega_{i,b[i]-1}$, and the increment in $\omega_{i,b[i]-1}$ is set 0 by Line 34 for satisfying WRTCs at $\mathrm{PS}_{i,b[i]-1}$.

After the above setting, WRTCs in each step are met. In addition, the increase of the next cycle is calculated by Lines 39 and 40 such that we can keep offsetting the increase in steps until $B < n[i] - 2$ does not hold. By Algorithm 6, we can offset the increase in the wafer sojourn time of each step to satisfy WRTCs.

Nevertheless, we need to make sure that two adjacent CTs can still coordinate well with each other to share their buffer chamber after the robot waiting time is set by Algorithm 6. We set Φ_1, Φ_2, Φ_3, and Φ_4 by Lines 3 and 4 in Algorithm 7. Line 13 of Algorithm 5 indicates $m_2^p + 2 \leq \kappa \leq i'$. Then, we have the following Theorem.

Theorem 6: After a cleaning operation, the robot waiting time is set by Algorithm 6, if the following (23) and (24) hold, two adjacent CTs still coordinate well to share their buffer chamber.

$$\sum_{f_2=1}^{b[i-1]-2} \sigma^{\kappa-1}_{i-1,f_2} + \sum_{f_5=b[i-1]}^{n[i-1]} \sigma^{\kappa}_{i-1,f_5} \geq \sigma^{\kappa}_{i,n[i]}, \ 2 \leq i \leq K \tag{23}$$

$$\sum_{f_8=1}^{n[i]-1} \sigma^{\kappa}_{i,f_8} \geq \sigma^{\kappa}_{i-1,b[i-1]-1}, \ 2 \leq i \leq K. \tag{24}$$

Proof: By (8), with Θ being the cycle time after the cleaning operation, the wafer sojourn time at $\mathrm{PS}_{i-1,b[i-1]}$ in the κth cycle is $\tau'_{i-1,b[i-1]} = \Theta - (4\lambda_{i-1} + 3\mu_{i-1} + \zeta^{\kappa}_{i-1,b[i-1]-1})$. Let ϕ_1 denote the time point when the loading operation of the $(\kappa-1)$th cycle at $\mathrm{PS}_{i-1,b[i-1]}$ ends, i.e., a wafer has been just loaded into a chamber at $\mathrm{PS}_{i-1,b[i-1]}$. In the κth cycle, let ϕ_2 denote the time point when robot R_{i-1} begins to move to $\mathrm{PS}_{i-1,b[i-1]}$. Then, $(\phi_2 - \phi_1) + \mu_{i-1} + \zeta^{\kappa}_{i-1,b[i-1]} = \tau'_{i-1,b[i-1]}$ is also the wafer sojourn time at $\mathrm{PS}_{i-1,b[i-1]}$ in the κth cycle. Starting at time point ϕ_1, by a backward strategy, the following operation sequence for R_{i-1} is performed: $\Gamma_1 = \langle$Moving to the chamber at $\mathrm{PS}_{i-1,b[i-1]-2}$ $(\mu_i) \rightarrow$ waiting there for $\zeta^{\kappa-1}_{i-1,b[i-1]-2}$ time units $\rightarrow$ unloading a finished wafer at $\mathrm{PS}_{i-1,b[i-1]-2}$ $(\lambda_i) \rightarrow$ moving to the chamber at $\mathrm{PS}_{i-1,b[i-1]-1}$ and loading it there $(\mu_i + \lambda_i) \rightarrow \cdots \rightarrow$ moving to $\mathrm{PS}_{i-1,0}$ $(\mu_i) \rightarrow$ waiting there for $\zeta^{\kappa}_{i-1,0}$ time units $\rightarrow$ unloading a wafer at $\mathrm{PS}_{i-1,0}$ $(\lambda_i) \rightarrow$ moving to the chamber at $\mathrm{PS}_{i-1,1}$ and loading it there $(\mu_i + \lambda_i) \rightarrow \cdots \rightarrow$ moving to $\mathrm{PS}_{i-1,b[i-1]+1}$ $(\mu_i) \rightarrow$ waiting there for $\zeta^{\kappa}_{i-1,b[i-1]+1}$ time units $\rightarrow$ unloading a finished wafer at $\mathrm{PS}_{i-1,b[i-1]+1}$ $(\lambda_i) \rightarrow$ moving to the chamber at $\mathrm{PS}_{i-1,b[i-1]+2}$ and loading it there $(\mu_i + \lambda_i) \rightarrow$ moving to the chamber at $\mathrm{PS}_{i-1,b[i-1]}\rangle$, and ψ' denotes the robot task time during such an operation sequence. Thus,

$$\begin{aligned}
\tau'_{i-1,b[i-1]} &= (\phi_2 - \phi_1) + \mu_{i-1} + \zeta^{\kappa}_{i-1,b[i-1]} \\
&= \Big(\sum_{e_1=1}^{b[i-1]-2} \zeta^{\kappa-1}_{i-1,e_1} + \sum_{e_2=b[i-1]+1}^{n[i-1]} \zeta^{\kappa}_{i-1,e_2} + \zeta^{\kappa}_{i-1,0} + \psi'\Big) + \mu_{i-1} \\
&\quad + \zeta^{\kappa}_{i-1,b[i-1]} \\
&= \omega_{i-1,0} + \sigma^{\kappa}_{i-1,0} + \sum_{f_1=1}^{b[i-1]-2} \omega_{i-1,f_1} + \sum_{f_2=1}^{b[i-1]-2} \sigma^{\kappa-1}_{i-1,f_2} \\
&\quad + \sum_{f_3=b[i-1]+1}^{n[i-1]} \omega_{i-1,f_3} + \sum_{f_4=b[i-1]+1}^{n[i-1]} \sigma^{\kappa}_{i-1,f_4} + \omega_{i-1,b[i-1]} \\
&\quad + \sigma^{\kappa}_{i-1,b[i-1]} + \mu_{i-1} + \psi' \\
&= \omega_{i-1,0} + \sum_{f_1=1}^{b[i-1]-2} \omega_{i-1,f_1} + \omega_{i-1,b[i-1]} \\
&\quad + \sum_{f_3=b[i-1]+1}^{n[i-1]} \omega_{i-1,f_3} + \mu_{i-1} \\
&\quad + \psi' + \sum_{f_2=1}^{b[i-1]-2} \sigma^{\kappa-1}_{i-1,f_2} + \sum_{f_5=b[i-1]}^{n[i-1]} \sigma^{\kappa}_{i-1,f_5} + \sigma^{\kappa}_{i-1,0} \\
&= \Big(\sum_{f_6=0}^{b[i-1]-2} \omega_{i-1,f_6} + \sum_{f_7=b[i-1]}^{n[i-1]} \omega_{i-1,f_7}\Big) + (\mu_{i-1} + \psi')
\end{aligned}$$

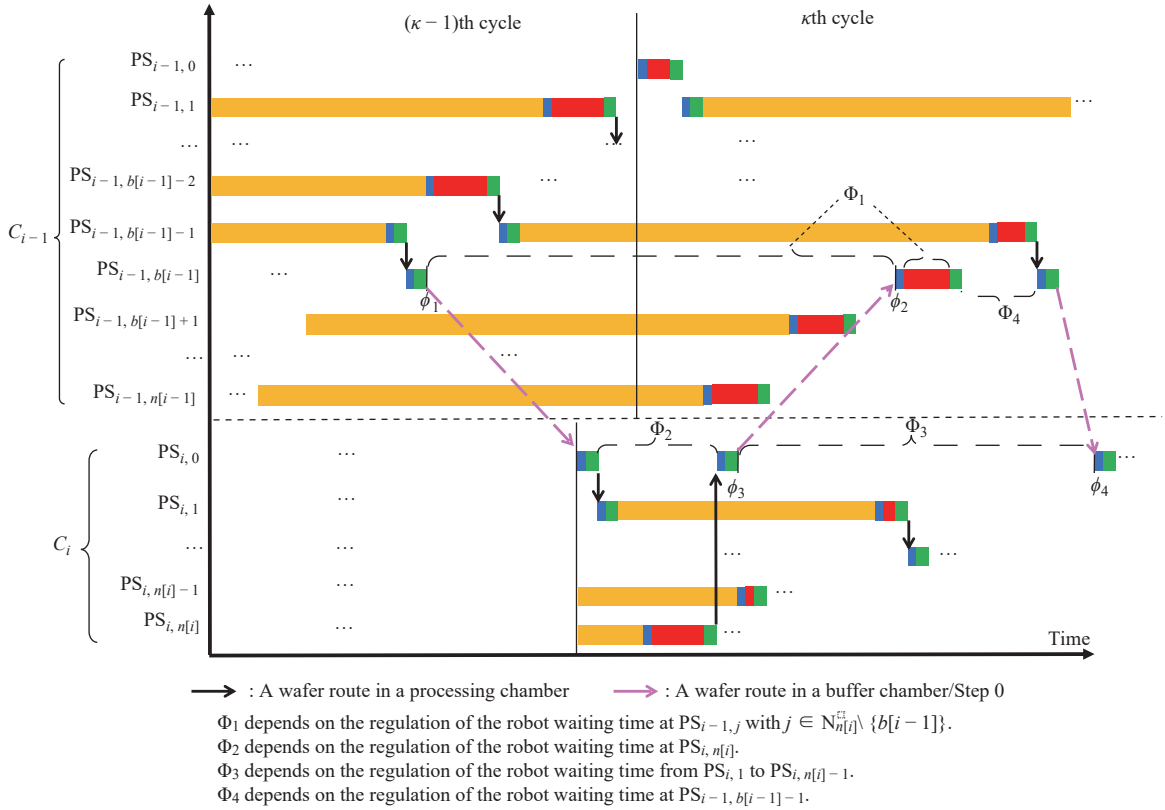

Fig. 5. Illustration of the coordination of a buffer chamber.

$$+\sum_{f_2=1}^{b[i-1]-2}\sigma^{\kappa-1}_{i-1,f_2}+\sum_{f_5=b[i-1]}^{n[i-1]}\sigma^{\kappa}_{i-1,f_5}+\sigma^{\kappa}_{i-1,0}$$

$$=(\psi_{i-1,2}-\omega_{i-1,b[i-1]-1})+(\psi_{i-1,1}-(4\lambda_{i-1}+3\mu_{i-1}))$$

$$+\sum_{f_2=1}^{b[i-1]-2}\sigma^{\kappa-1}_{i-1,f_2}+\sum_{f_5=b[i-1]}^{n[i-1]}\sigma^{\kappa}_{i-1,f_5}+\sigma^{\kappa}_{i-1,0}$$

$$=\psi_{i-1,2}+\psi_{i-1,1}-(4\lambda_{i-1}+3\mu_{i-1}+\omega_{i-1,b[i-1]-1})$$

$$+\sum_{f_2=1}^{b[i-1]-2}\sigma^{\kappa-1}_{i-1,f_2}+\sum_{f_5=b[i-1]}^{n[i-1]}\sigma^{\kappa}_{i-1,f_5}+\sigma^{\kappa}_{i-1,0}$$

$$=\Theta-(4\lambda_{i-1}+3\mu_{i-1}+\omega_{i-1,b[i-1]-1})+\sum_{f_2=1}^{b[i-1]-2}\sigma^{\kappa-1}_{i-1,f_2}$$

$$+\sum_{f_5=b[i-1]}^{n[i-1]}\sigma^{\kappa}_{i-1,f_5}+\sigma^{\kappa}_{i-1,0}.$$

By Algorithm 5, $\sigma^{\kappa}_{i-1,0}$ is always 0, and it follows from (8) that $\Theta-(4\lambda_{i-1}+3\mu_{i-1}+\omega_{i-1,b[i-1]-1})+\sum_{f_2=1}^{b[i-1]-2}\sigma^{\kappa-1}_{i-1,f_2}+\sum_{f_5=b[i-1]}^{n[i-1]}\sigma^{\kappa}_{i-1,f_5}+\sigma^{\kappa}_{i-1,0}$ becomes:

$$\tau'_{i-1,b[i-1]}=\tau_{i-1,b[i-1]}+\Phi_1$$

where $\Phi_1=\sum_{f_2=1}^{b[i-1]-2}\sigma^{\kappa-1}_{i-1,f_2}+\sum_{f_5=b[i-1]}^{n[i-1]}\sigma^{\kappa}_{i-1,f_5}$. Set $\Phi_2=\sigma^{\kappa}_{i,n[i]}$, which is always a decrement in the robot waiting time $\omega_{i,n[i]}$ by Lines 5 and 8 in Algorithm 6. If (23) holds, it means that

$$\tau'_{i-1,b[i-1]}=\tau_{i-1,b[i-1]}+\Phi_1\geq 4\lambda_i+3\mu_i+\omega_{i,n[i]}+\Phi_2$$

where $\tau_{i-1,b[i-1]}\geq 4\lambda_i+3\mu_i+\omega_{i,n[i]}$, or (13) is already satisfied by the previous analysis.

In the κth cycle, let ϕ_3 denote the time point when the loading operation at $PS_{i,0}$ ends, i.e., R_i has just loaded a wafer into a chamber at $PS_{i,0}$, and ϕ_4 the time point when the moving operation starts, i.e., R_i begins to move to $PS_{i,0}$. Then, $(\phi_4-\phi_3)+\mu_i+\zeta^{\kappa}_{i,0}=\tau'_{i,0}$ also is the wafer sojourn time at $PS_{i,0}$ in the κth cycle. Similarly,

$$\tau'_{i,0}=\tau_{i,0}+\Phi_3$$

where $\Phi_3=\sum_{j=1}^{n[i]-1}\sigma_{i,j}$. In addition, $\Phi_4=\sigma_{i-1,b[i-1]-1}$. If (24) holds, it means that

$$\begin{aligned}\tau'_{i,0}&=\tau_{i,0}+\Phi_3\\&\geq 4\lambda_{i-1}+3\mu_{i-1}+\omega_{i-1,b[i-1]-1}+\Phi_4\\&=4\lambda_{i-1}+3\mu_{i-1}+\zeta^{\kappa}_{i-1,b[i-1]-1}\end{aligned}$$

namely, (14) is satisfied.

Therefore, if (23) and (24) hold, then (13) and (14) are satisfied, which means that two adjacent CTs can well coordinate to share their buffer chamber by Algorithm 6. ■

The above analysis is shown in Fig. 5. It shows an example of a pair of adjacent tools coordinating to share their buffer chamber when (23) and (24) hold.

In Algorithm 6, the number of iterations in the For-loop of Lines 8−27 and 37−38 is no more than $n[i]$. Within Lines 8−27, there are several summations, which show that the number of iterations of such functions is also no more than $n[i]$. The running time of the outer DoWhile-loop in Algorithm 6 is $(n[i])^2\times\iota$, where ι is the number of iterations in the DoWhile-

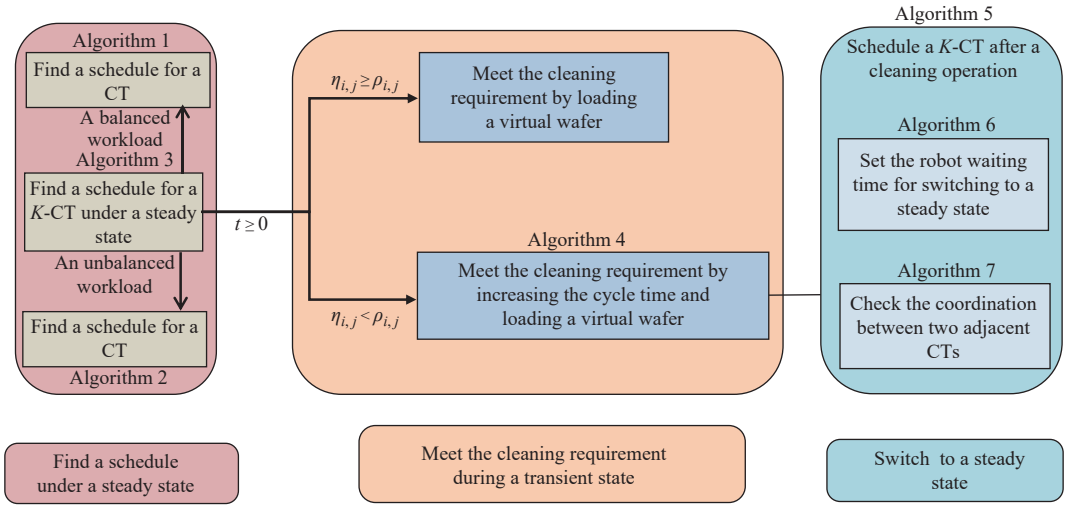

Fig. 6. Overview of the proposed algorithms.

loop. The number of iterations depends on the number of parallel chambers at the last step ($m^i_{n[i]}$), which is a constant. Therefore, the computational complexity of Algorithm 6 is polynomial.

Then, in Algorithm 7, the number of operations in summations is no more than $n[i]$, which is a constant. Thus, the computational complexity of Algorithm 7 is polynomial.

Algorithm 5 calls Algorithms 6 and 7 to set the robot waiting time and check the coordination between two adjacent CTs. Thus, the computational complexity of Algorithm 5 depends on the complexity of Algorithms 6 and 7. By Lines 13 and 14 in Algorithm 5, the iteration number of Algorithm 7 is $\iota' - (m^p_2 + 2)$, which depends on the number of iterations in the DoWhile-loop within Algorithm 6. Similarly, ι' depends on $m^i_{n[i]}$ like ι in Algorithm 6, which is a constant. The number of iterations in the For-loop of Lines 2−14 in Algorithm 5 is no more than K, which is a constant. Therefore, the running time of the outer For-loop in Algorithm 5 is $K \times (((n[i])^2 \times \iota) + ((n[i]) \times (\iota' - (m^p_2 + 2)))$. In practice, a CT consists of four to eight PCs, which indicates that $n[i]$ and m^i_j, $i \in \mathrm{N}^+_K, j \in \mathrm{N}^+_{n[i]}$ are limited and small. Since ι and ι' depend on $m^i_{n[i]}$ according to the above analysis, the computational complexity of Algorithm 5 is polynomial.

E. Summary of Algorithms

Within a K-CT, Algorithm 3 checks the workload for each CT and calls Algorithm 1 or Algorithm 2 to give the schedulability and set the robot waiting time if such a CT is schedulable. By the obtained robot waiting time, a feasible schedule can be found for a K-CT with WRTCs and parallel chambers during a steady state.

When the controller sends the clean signal, we calculate t by (17) for determining the time to unload a virtual wafer from LLs in order to load it into the cleaning chamber on time. If $t \geq 0$ and $\eta_{i,j} \geq \rho_{i,j}$, it means that the time for cleaning gained by loading a virtual wafer is longer than the time for finishing such a cleaning operation. Since a virtual wafer is fictitious, after loading it into the cleaning chamber, the robot just waits here until its next operation, which indicates that the robot operation sequences remain unchanged. And the cleaning operation can be performed completely without interruption until the next loading operation occurs at the cleaning chamber.

If $t > 0$ and $\eta_{i,j} < \rho_{i,j}$, it means that the time gained for cleaning by loading a virtual wafer is not enough for meeting the cleaning requirement when such a cleaning operation lasts a longer time. Under this circumstance, if (19)−(22) are satisfied, we can increase the cycle time by Algorithm 4 to gain enough time for cleaning. Consequently, the coordination between two adjacent CTs is still valid, and WRTCs are still satisfied. After the cleaning operation is finished, Algorithm 5 is adopted to switch a K-CT to a steady state. Algorithm 5 calls Algorithm 6 to adjust the robot waiting time satisfying WRTCs for each CT so that such waiting time is the same as that in a steady state. Then, Algorithm 7 checks whether the coordination between two adjacent CTs still holds after such adjustments. If such coordination still holds, it returns $Q_1 = 0$, and a K-CT is back to a steady state. The above analysis can be summarized in Fig. 6.

IV. Illustrative Examples

This section uses two examples to show the application of the proposed algorithms. In the following examples, the time unit is second.

Example 1: In a three-CT, $\mathrm{PS}_{1,0}$ is LLs, $\mathrm{PS}_{2,0,}$ and $\mathrm{PS}_{3,0}$ are incoming buffers, and, $\mathrm{PS}_{1,2}$ and $\mathrm{PS}_{2,3}$ are outgoing buffers. Each robot takes 2 s to move between two steps. The time taken by a robot for loading/unloading operation is 3 s. Or we have $\lambda_1 = \lambda_2 = \lambda_3 = 3$ and $\mu_1 = \mu_2 = \mu_3 = 2$. C_1 has three processing steps with $(m^1_1, m^1_{b[1]}, m^1_3) = (2, 1, 2)$, where $b[i] = 2$. The wafer processing time at Steps 1−3 is 100, 0, and 80, respectively, or we have $\alpha_{1,1} = 100$, $\alpha_{1,2} = 0$, and $\alpha_{1,3} = 80$. After being processed, a wafer has to be unloaded from a PC within no more than 20 s, or we have $\delta_{1,1} = \delta_{1,3} = 20$. There are four processing steps within C_2, which means that the wafer flow is $(m^2_1, m^2_2, m^2_{b[2]}, m^2_4) = (3, 3, 1, 2)$, where $b[i] = 3$. The wafer processing time at Steps 1−4 is 174, 180, 0, and 100 respectively, or we have $\alpha_{2,1} = 174$, $\alpha_{2,2} = 180$, $\alpha_{2,3} = 0$, and $\alpha_{2,4} = 100$. Similarly, we have $\delta_{2,1} = 20$, $\delta_{2,2} = 20$, and

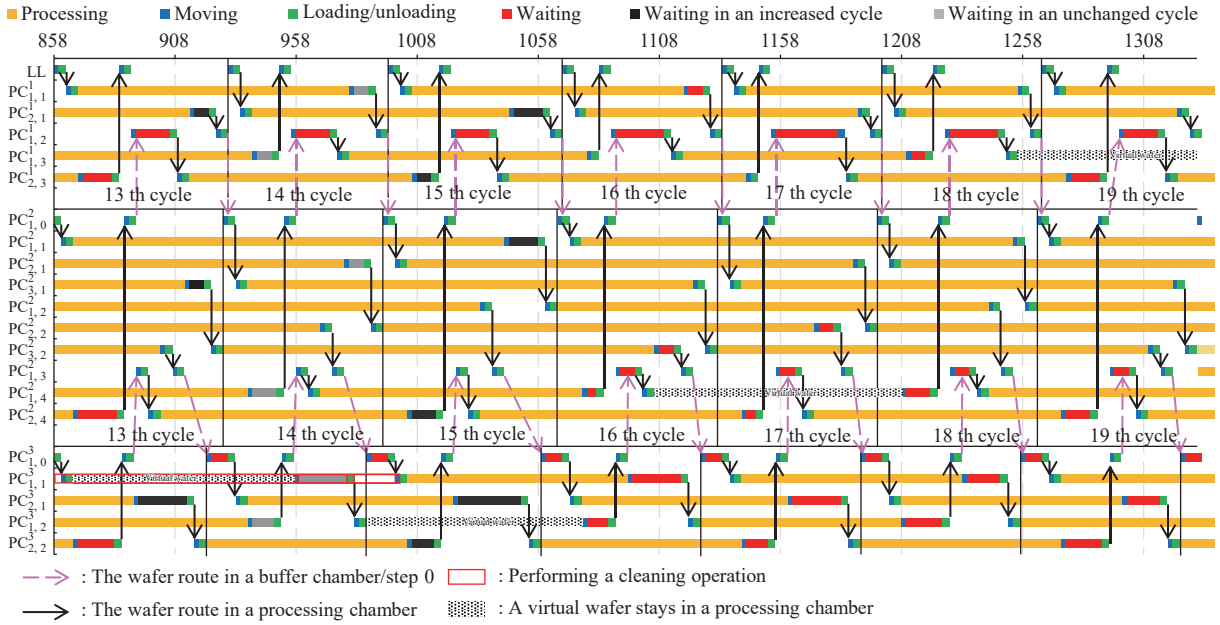

Fig. 7. A three-CT with a condition-based cleaning operation.

$\delta_{2,4} = 20$. For C_3 with $(m_1^3, m_2^3) = (2, 2)$, the wafer processing time at Steps 1 and 2 is 100 and 80, or we have $\alpha_{3,1} = 100$ and $\alpha_{3,2} = 80$. The wafer can stay there for 20 s after being processed, or $\delta_{3,1} = \delta_{3,2} = 20$.

In such a case, during a steady state, the cycle time is found to be $\Theta = 66$. It can be verified that 1) C_1 and C_3 satisfy the condition in Case 2; 2) C_2 satisfies the condition in Case 1. Then, by Algorithm 3, in a steady state, the robot waiting time is set as $\omega_{1,0} = \omega_{1,1} = 0$, $\omega_{1,2} = 14$, $\omega_{1,3} = 12$, $\omega_{2,0} = \omega_{2,1} = \omega_{2,2} = \omega_{2,3} = 0$, $\omega_{2,4} = 16$, $\omega_{3,0} = 7$, $\omega_{3,1} = 14$, and $\omega_{3,2} = 15$. Thus, a feasible schedule can be determined for such a three-CT.

By the controller of such a three-CT, in order to ensure the wafer quality, the chamber ($PC_{1,1}^3$) needs to be cleaned after six wafers are consecutively processed. Therefore, the time point that unloads a virtual wafer from LLs can be calculated by (17), which means $t = (13 - 3 - 7) \times \Theta + \mu_3 + \omega_{3,0} = 669 > 0$. A virtual wafer can be loaded into a chamber ($PC_{1,1}^3$) on time for its cleaning. The cleaning operation takes 157s, which shows $\rho_{3,1} > \eta_{3,1} = \Theta \times m_1^3 + 3\mu_3 + 2\lambda_3 + \omega_{3,0} = 151$. It means that the cleaning time obtained by loading a virtual wafer is not enough to complete the cleaning operation. Hence, by Algorithm 4, we increase the cycle time intermittently by $\Delta = (\rho_{3,1} - \eta_{3,1})/\, x_0^3 = 6$.

During the cleaning operation, by Algorithm 4, the robot waiting time is set as follows.

1) In the 13th cycle, $\omega_{1,0}^1 = 0$, $\omega_{1,1}^1 = 6$, $\omega_{1,2}^1 = 14$, $\omega_{1,3}^1 = 12$, $\omega_{2,0}^1 = 0$, $\omega_{2,1}^1 =6$, $\omega_{2,2}^1 = \omega_{2,3}^1 = 0$, $\omega_{2,4}^1 = 16$, $\omega_{3,0}^1 = 7$, $\omega_{3,1}^1 = 20$, and $\omega_{3,2}^1 = 15$;

2) In the 14th cycle, $\omega_{1,0}^2 = 0$, $\omega_{1,1}^2 = 6$, $\omega_{1,2}^2 = 14$, $\omega_{1,3}^2 = 6$, $\omega_{2,0}^2 = 0$, $\omega_{2,1}^2 =6$, $\omega_{2,2}^2 = \omega_{2,3}^2 = 0$, $\omega_{2,4}^2 = 10$, $\omega_{3,0}^2 = 7$, $\omega_{3,1}^2 = 20$, and $\omega_{3,2}^2 = 9$;

3) In the 15th cycle, $\omega_{1,0}^3 = 0$, $\omega_{1,1}^3 = 12$, $\omega_{1,2}^3 = 14$, $\omega_{1,3}^3 = 6$, $\omega_{2,0}^3 = 0$, $\omega_{2,1}^3 =12$, $\omega_{2,2}^3 = \omega_{2,3}^3 = 0$, $\omega_{2,4}^3 = 10$, $\omega_{3,0}^3 = 7$, $\omega_{3,1}^3 = 26$, and $\omega_{3,2}^3 = 9$.

Therefore, the three-CT can obtain enough cleaning time to finish the cleaning operation performed at $PC_{1,1}^3$.

After the cleaning operation, by Algorithm 5 with $Q_1 = 0$, the robot waiting time is set as:

4) In the 16th cycle, $\zeta_{1,0}^4 = 0$, $\zeta_{1,1}^4 = 6$, $\zeta_{1,2}^4 = 20$, $\zeta_{1,3}^4 = 0$, $\zeta_{2,0}^4 = 0$, $\zeta_{2,1}^4 = 6$, $\zeta_{2,2}^4 = 0$, $\zeta_{2,3}^4 = 6$, $\zeta_{2,4}^4 = 4$, $\zeta_{3,0}^4 = 7$, $\zeta_{3,1}^4 = 20$, and $\zeta_{3,2}^4 = 9$;

5) In the 17th cycle, $\zeta_{1,0}^5 = 0$, $\zeta_{1,1}^5 = 0$, $\zeta_{1,2}^5 = 26$, $\zeta_{1,3}^5 = 0$, $\zeta_{2,0}^5 = 0$, $\zeta_{2,1}^5 = 6$, $\zeta_{2,2}^5 = 0$, $\zeta_{2,3}^5 = 6$, $\zeta_{2,4}^5 = 4$, $\zeta_{3,0}^5 = 7$, $\zeta_{3,1}^5 = 20$, and $\zeta_{3,2}^5 = 9$;

6) In the 18th cycle, $\zeta_{1,0}^6 = 0$, $\zeta_{1,1}^6 = 0$, $\zeta_{1,2}^6 = 20$, $\zeta_{1,3}^6 = 6$, $\zeta_{2,0}^6 = 0$, $\zeta_{2,1}^6 = 0$, $\zeta_{2,2}^6 = 0$, $\zeta_{2,3}^6 = 6$, $\zeta_{2,4}^6 = 10$, $\zeta_{3,0}^6 = 7$, $\zeta_{3,1}^6 = 14$, and $\zeta_{3,2}^6 = 15$, and C_3 returns to the steady state;

7) In the 19th cycle, $\zeta_{1,0}^7 = 0$, $\zeta_{1,1}^7 = 0$, $\zeta_{1,2}^7 = 14$, $\zeta_{1,3}^7 = 12$, $\zeta_{2,0}^6 = 0$, $\zeta_{2,1}^7 = 0$, $\zeta_{2,2}^7 = 0$, $\zeta_{2,3}^7 = 6$, $\zeta_{2,4}^7 = 10$, and C_1 returns to the steady state, while C_2 can return to a steady state in the 20th cycle. It is verified in a Gantt chart shown in Fig. 7.

Example 2: In a two-CT, $PS_{1,0}$ is LLs, $PS_{2,0}$ is an incoming buffer, and $PS_{1,2}$ is an outgoing buffer. R_1 takes 3 s to move between two steps, while R_2 takes 2 s to do so. The time taken by R_1 for the loading/unloading operation is 2 s. R_2 takes 1 s to load/unload a wafer. Namely, we have $\lambda_1 = 2$, $\lambda_2 = 1$ and $\mu_1 = 3$, $\mu_2 = 2$. C_1 has three processing steps with $(m_1^1, m_{b[1]}^1, m_3^1) = (3, 1, 2)$, where $b[i] = 2$. The wafer processing time at Steps 1–3 is 154, 0, and 93, respectively, or we have $\alpha_{1,1} = 154$, $\alpha_{1,2} = 0$, and $\alpha_{1,3} = 93$. After being processed, a wafer has to be unloaded from a PC within no more than 20 s, or we have $\delta_{1,1} = \delta_{1,3} = 20$. There are two processing steps within C_2, which means that the wafer flow is $(m_1^2, m_2^2) = (3, 3)$. The wafer processing time at Steps 1 and 2 is 152, and 127 respectively, or we have $\alpha_{2,1} = 152$, and $\alpha_{2,2} = 127$. Similarly, we have $\delta_{2,1} = 20$, and $\delta_{2,2} = 20$.

In such a case, during a steady state, the cycle time is found to be $\Theta = 57$. It can be verified that 1) C_1 satisfies the condi-

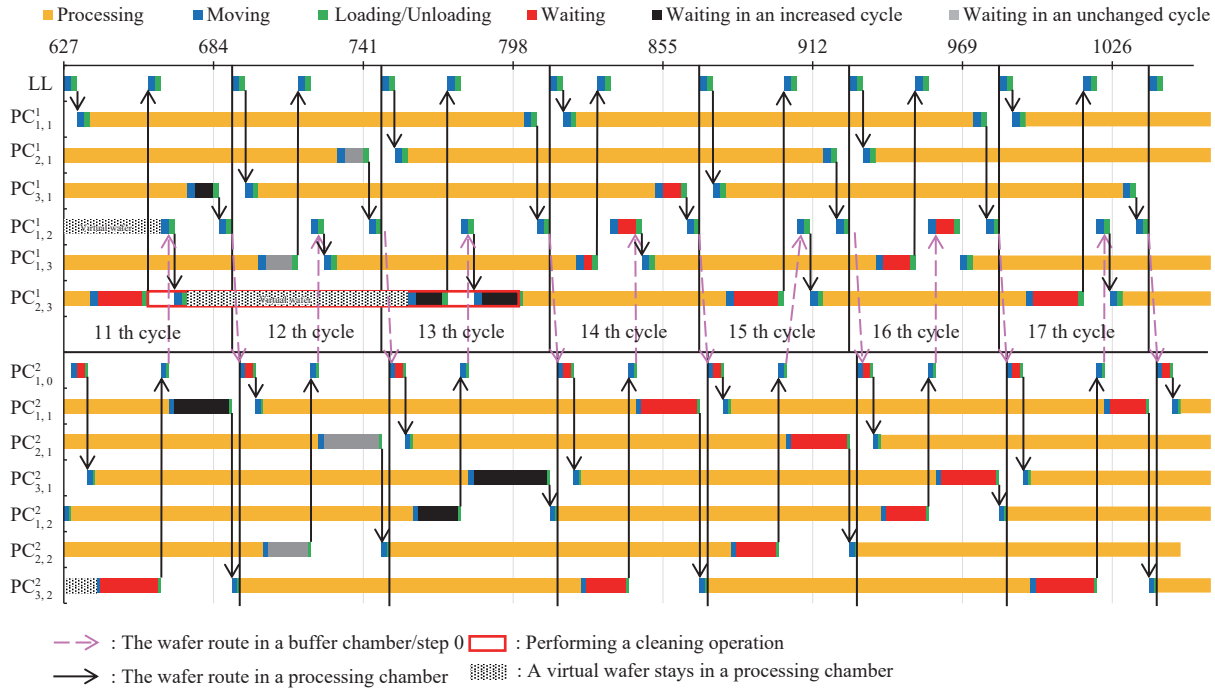

Fig. 8. A two-CT with a condition-based cleaning operation.

tions in Case 1; 2) C_2 satisfies the conditions in Case 2. Then, by Algorithm 3, in a steady state, the robot waiting time is set as $\omega_{1,0} = \omega_{1,1} = \omega_{1,2} = 0$, $\omega_{1,3} = 17$, $\omega_{2,0} = 3$, $\omega_{2,1} = 14$, and $\omega_{2,2} = 22$. Thus, a feasible schedule can be determined for such a two-CT.

By the controller of such a two-CT, in order to ensure the wafer quality, the chamber ($PC^1_{2,3}$) needs to be cleaned after five wafers are consecutively processed. Therefore, the time point that R_1 unloads a virtual wafer from LLs can be calculated by (17), which means $t = (11 - 9) \times \Theta + \mu_1 + \omega_{1,0} = 117 > 0$. A virtual wafer can be loaded into a chamber ($PC^1_{2,3}$) on time for its cleaning. The cleaning operation takes 141 s, which shows $\rho_{1,3} > \eta_{1,3} = \Theta \times m_3^1 + 3\mu_1 + 2\lambda_1 + \omega_{1,2} = 127$. It means that the cleaning time obtained by loading a virtual wafer is not enough to complete the cleaning operation. Hence, by Algorithm 4, we increase the cycle time intermittently by $\Delta = (\rho_{1,3} - \eta_{1,3}) / (\lfloor \frac{m_2^p}{2} \rfloor + 1) = 14 / 2 = 7$.

During the cleaning operation, by Algorithm 4, the robot waiting time is set as follows.

1) In the 11th cycle, $\omega^1_{1,0} = 0$, $\omega^1_{1,1} = 7$, $\omega^1_{1,2} = 0$, $\omega^1_{1,3} = 17$, $\omega^1_{2,0} = 3$, $\omega^1_{2,1} = 21$, $\omega^1_{2,2} = 22$;

2) In the 12th cycle, $\omega^2_{1,0} = 0$, $\omega^2_{1,1} = 7$, $\omega^2_{1,2} = 0$, $\omega^2_{1,3} = 10$, $\omega^2_{2,0} = 3$, $\omega^2_{2,1} = 21$, $\omega^2_{2,2} = 15$;

3) In the 13th cycle, $\omega^3_{1,0} = 0$, $\omega^3_{1,1} = 0$, $\omega^3_{1,2} = 0$, $\omega^3_{1,3} = 10$, $\gamma_{1,3} = 14$, $\omega^3_{2,0} = 3$, $\omega^3_{2,1} = 21$, $\omega^3_{2,2} = 22$.

Therefore, the two-CT can obtain enough cleaning time to finish the cleaning operation performed at $PC^1_{2,3}$.

After the cleaning operation, by Algorithm 5 with $Q_1 = 0$, the robot waiting time is set as:

4) In the 14th cycle, $\zeta^4_{1,0} = 0$, $\zeta^4_{1,1} = 7$, $\zeta^4_{1,2} = 7$, $\zeta^4_{1,3} = 3$, $\zeta^4_{2,0} = 3$, $\zeta^4_{2,1} = 21$, $\zeta^4_{2,2} = 15$;

5) In the 15th cycle, $\zeta^5_{1,0} = 0$, $\zeta^5_{1,1} = 0$, $\zeta^5_{1,2} = 0$, $\zeta^5_{1,3} = 17$, $\zeta^5_{2,0} = 3$, $\zeta^5_{2,1} = 21$, $\zeta^5_{2,2} = 15$;

6) In the 16th cycle, $\zeta^6_{1,0} = 0$, $\zeta^6_{1,1} = 0$, $\zeta^6_{1,2} = 7$, $\zeta^6_{1,3} = 10$, $\zeta^6_{2,0} = 3$, $\zeta^6_{2,1} = 21$, $\zeta^6_{2,2} = 15$. Then, C_1 and C_2 can switch to a steady state in the next cycle. It is verified in a Gantt chart shown in Fig. 8.

V. Conclusions

This study deals with a scheduling problem for a single-arm MCT with WRTCs and a condition-based cleaning operation which is commonly seen in semiconductor manufacturing. Differently from the studies on scheduling a single-arm CT with a purge operation, this study focuses on a more general case in which a chamber has finished more than one wafer before a cleaning operation is required. To do so, this work presents a virtual-wafer-based method such that the robot loads either a real wafer or a virtual wafer at a time point t before the cleaning operation starts. Then, based on the schedule under a steady state, by using a virtual wafer, we can find a feasible schedule for a K-CT running with the condition-based cleaning operation and satisfying WRTCs. Furthermore, for a condition-based cleaning operation with a long cleaning time, we can implement the same method by increasing the cycle time to complete such a cleaning operation. Besides, the obtained solution is a periodic schedule based on a one-wafer cyclic schedule that can be easily applied. Therefore, it has high practical value in semiconductor manufacturing.

In future work, we will investigate how to schedule a multi-cluster tool when two or more cleaning operations are performed simultaneously, which is different from this work.

References

[1] M. Dawande, C. Sriskandarajah, and S. Sethi, "On throughput maximization in constant travel-time robotic cells," *Manufact. Serv.*

Oper. Manage., vol. 4, no. 4, pp. 296–312, Oct. 2002.

[2] M. J. López and S. C. Wood, "Systems of multiple cluster tools: Configuration, reliability, and performance," *IEEE Trans. Semicond. Manuf.*, vol. 16, no. 2, pp. 170–178, May 2003.

[3] T.-E. Lee, H.-Y. Lee, and Y.-H. Shin, "Workload balancing and scheduling of a single-armed cluster tool," in *Proc. 5th APIEMS Conf.*, 2004, pp. 1–15.

[4] T.-E. Lee, "A review of scheduling theory and methods for semiconductor manufacturing cluster tools," in *Proc. Winter Simulation Conf.*, 2008, pp. 2127–2135.

[5] H. N. Geismar, M. Pinedo, and C. Sriskandarajah, "Robotic cells with parallel machines and multiple dual gripper robots: A comparative overview," *IIE Trans.*, vol. 40, no. 12, pp. 1211–1227, Oct. 2008.

[6] H.-J. Kim and J.-H. Lee, "Closed-form expressions on lot completion time for dual-armed cluster tools with parallel processing modules," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 898–907, Apr. 2019.

[7] S. Rostami, B. Hamidzadeh, and D. Camporese, "An optimal periodic scheduler for dual-arm robots in cluster tools with residency constraints," *IEEE Trans. Robot. Autom.*, vol. 17, no. 5, pp. 609–618, Oct. 2001.

[8] Y.-H. Shin, T.-E. Lee, J.-H. Kim, and H.-Y. Lee, "Modeling and implementing a real-time scheduler for dual-armed cluster tools," *Comput. Ind.*, vol. 45, no. 1, pp. 13–27, May 2001.

[9] D.-K. Kim, H.-J. Kim, and T.-E. Lee, "Optimal scheduling for sequentially connected cluster tools with dual-armed robots and a single input and output module," *Int. J. Prod. Res.*, vol. 55, no. 11, pp. 3092–3109, Jun. 2017.

[10] W. Kim, T. S. Yu, and T. E. Lee, "Integrated scheduling of a dual-armed cluster tool for maximizing steady schedule patterns," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 51, no. 12, pp. 7282–7294, Dec. 2021.

[11] J. P. Wang, H. S. Hu, C. R. Pan, Y. Zhou, and L. Li, "Scheduling dual-arm cluster tools with multiple wafer types and residency time constraints," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 3, pp. 776–789, May 2020.

[12] B. Huang and M. Zhou, "Symbolic scheduling of robotic cellular manufacturing systems with timed Petri Nets," *IEEE Trans. Control Syst. Technol.*, vol. 30, no. 5, pp. 1876–1887, Sept. 2022.

[13] J. Luo, Z. Liu, M. Zhou, and K. Xing, "Deadlock-free scheduling of flexible assembly systems based on Petri Nets and local search," *IEEE Trans. Syst. Man Cybern.:Syst.*, vol. 50, no. 10, pp. 3658–3669, Oct. 2020.

[14] J. Luo, M. Zhou, and J. Q. Wang, "AB&B: An anytime branch and bound algorithm for scheduling of deadlock-prone flexible manufacturing systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 4, pp. 2011–2021, Oct. 2021.

[15] S. Wang, X. Guo, O. Karoui, M. Zhou, D. You, and A. Abusorrah, "A refined siphon-based deadlock prevention policy for a class of Petri Nets," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 53, no. 1, pp. 191–203, Jan. 2023.

[16] J.-H. Lee and H.-J. Kim, "Backward sequence analysis for single-armed cluster tools," *ICAPS*, vol. 29, no. 1, pp. 269–272, May 2021.

[17] Q. H. Zhu, Y. Qiao, N. Q. Wu, and Y. Hou, "Post-processing time-aware optimal scheduling of single robotic cluster tools," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 2, pp. 597–605, Mar. 2020.

[18] N. Q. Wu, C. B. Chu, F. Chu, and M. C. Zhou, "A Petri Net method for schedulability and scheduling problems in single-arm cluster tools with wafer residency time constraints," *IEEE Trans. Semicond. Manuf.*, vol. 21, no. 2, pp. 224–237, May 2008.

[19] C. R. Pan, Y. Qiao, N. Q. Wu, and M. C. Zhou, "A novel algorithm for wafer sojourn time analysis of single-arm cluster tools with wafer residency time constraints and activity time variation," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 45, no. 5, pp. 805–818, May 2015.

[20] F. J. Yang, N. Q. Wu, Y. Qiao, M. C. Zhou, and Z. W. Li, "Scheduling of single-arm cluster tools for an atomic layer deposition process with residency time constraints," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 47, no. 3, pp. 502–516, Mar. 2017.

[21] F. Yang, K. Gao, I. W. Simon, Y. Zhu, and R. Su, "Decomposition methods for manufacturing system scheduling: A survey," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 2, pp. 389–400, Mar. 2018.

[22] F. J. Yang, X. Tang, N. Q. Wu, C. J. Zhang, and L. Gao, "Wafer residency time analysis for time-constrained single-robot-arm cluster tools with activity time variation," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 4, pp. 1177–1188, Jul. 2020.

[23] Q. H. Zhu, N. Q. Wu, Y. Qiao, and M. C. Zhou, "Petri Net-based optimal one-wafer scheduling of single-arm multi-cluster tools in semiconductor manufacturing," *IEEE Trans. Semicond. Manuf.*, vol. 26, no. 4, pp. 578–591, Feb. 2013.

[24] F. Yang, N. Wu, Y. Qiao, M. Zhou, R. Su, and T. Qu, "Petri Net-based efficient determination of optimal schedules for transport-dominant single-arm multi-cluster tools," *IEEE Access*, vol. 6, pp. 355–365, Nov. 2017.

[25] X. Li and R. Y. K. Fung, "Optimal K-unit cycle scheduling of two-cluster tools with residency constraints and general robot moving times," *J. Scheduling*, vol. 19, no. 2, pp. 165–176, Apr. 2016.

[26] L. P. Bai, N. Q. Wu, Z. W. Li, and M. C. Zhou, "Optimal one-wafer cyclic scheduling and buffer space configuration for single-arm multicluster tools with linear topology," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 46, no. 10, pp. 1456–1467, Oct. 2016.

[27] H.-J. Kim, J.-H. Lee, C. Kim, S. H. Baik, and T.-E. Lee, "Scheduling in-line mulitple cluster tools: A decomposition approach," in *Proc. Int. Conf. Mech. and Autom.*, 2012, pp. 1544–1549.

[28] F. J. Yang, N. Q. Wu, Y. Qiao, and R. Su, "Polynomial approach to optimal one-wafer cyclic scheduling of treelike hybrid multi-cluster tools via Petri Nets," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 1, pp. 270–280, Jan. 2018.

[29] Q. H. Zhu, G. H. Wang, Y. Hou, N. Q. Wu, and Y. Qiao, "Optimally scheduling dual-arm multi-cluster tools to process two wafer types," *IEEE Rob. Autom. Lett.*, vol. 7, no. 3, pp. 5920–5927, Jul. 2022.

[30] H.-Y. Jin and J. R. Morrison, "Transient scheduling of single armed cluster tools: Algorithms for wafer residency constraints," in *Proc. IEEE Int. Conf. Autom. Sci. Eng.*, 2013, pp. 856–861.

[31] Q. H. Zhu, Y. Qiao, and N. Q. Wu, "Optimal integrated schedule of entire process of dual-blade multi-cluster tools from start-up to close-down," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 2, pp. 553–565, Mar. 2019.

[32] Q. H. Zhu, M. C. Zhou, Y. Qiao, and N. Q. Wu, "Petri Net modeling and scheduling of a close-down process for time-constrained single-arm cluster tools," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 48, no. 3, pp. 389–400, Mar. 2018.

[33] T.-K. Kim, C. Junga, and T.-E. Lee, "Scheduling start-up and close-down periods of dual-armed cluster tools with wafer delay regulation," *Int. J. Prod. Res.*, vol. 50, no. 10, pp. 2785–2795, May 2012.

[34] D.-K. Kim, T.-E. Lee, and H.-J. Kim, "Optimal scheduling of transient cycles for single-armed cluster tools with parallel chambers," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 1165–1175, Apr. 2016.

[35] Y. Ahn and J. R. Morrison, "Analysis of circular cluster tools: Transient behavior and semiconductor equipment models," in *Proc. IEEE Int. Conf. Autom. Sci. Eng.*, 2010, pp. 39–44.

[36] F. F. Wang, F. Ju, and N. X. Kang, "Transient analysis and real-time control of geometric serial lines with residence time constraints," *IISE Trans.*, vol. 51, no. 7, pp. 709–728, Jul. 2019.

[37] D.-K. Kim, T.-E. Lee, and H.-J. Kim, "Optimal scheduling of transient cycles for single-armed cluster tools," in *Proc. IEEE Int. Conf. Autom. Sci. Eng.*, 2013, pp. 874–879.

[38] T.-S. Yu, H.-J. Kim, and T.-E. Lee, "Scheduling single-armed cluster tools with chamber cleaning operations," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 705–716, Apr. 2018.

[39] T.-S. Yu and T.-E. Lee, "Scheduling dual-armed cluster tools with chamber cleaning operations," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 1, pp. 218–228, Jan. 2019.

[40] F. J. Yang, K. Z. Gao, C. J. Zhang, Y. T. Zhu, R. Su, and Y. Qiao, "Efficient approach to cyclic scheduling of single-arm cluster tools with chamber cleaning operations and wafer residency time constraint,"

IEEE Trans. Semicond. Manuf., vol. 31, no. 2, pp. 196–205, May 2018.

[41] M. Lee, J. R. Morrison, and A. A. Kalir, "Practical queueing models for preventive maintenance plan optimization: Multiple maintenance types and numerical studies," *IEEE Trans. Semicond. Manuf.*, vol. 34, no. 1, pp. 104–114, Dec. 2021.

[42] S.-M. Noh, J.-H. Kim, and S.-Y. Jang, "A schdule of cleaning process for a single-armed cluster tool," *Int. J. Industrial Engineering*, vol. 24, no. 2, pp. 232–244, Jun. 2017.

[43] C. Hong and T.-E. Lee, "Multi-agent reinforcement learning approach for scheduling cluster tools with condition based chamber cleaning operations," in *Proc. IEEE Int. Conf. Mach. Learn. Appl.*, 2018, pp. 885–890.

[44] C. Li, F. Yang, and L. Zhen, "Efficient scheduling approaches to time-constrained single-armed cluster tools with condition-based chamber cleaning operations," *Int. J. Prod. Res.*, vol. 60, no. 11, pp. 3555–3568, Jun. 2022.

[45] J. Li, Y. Qiao, S. Zhang, Z. Li, N. Wu, and T. Song, "Scheduling of single-arm cluster tools with residency time constraints and chamber cleaning operations," *Appl. Sciences*, vol. 11, no. 19, pp. 9193–9211, Oct. 2021.

Qinghua Zhu (Senior Member, IEEE) received the Ph.D. degree in industrial engineering from Guangdong University of Technology in 2013. From 2014 to 2015 and from 2017 to 2018, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, USA. He joined Guangdong University of Technology in 2003, and is currently an Associate Professor of the School of Computer Science and Technology. His research interests include intelligent manufacturing scheduling, cloud computing, edge computing, discrete event systems, and Petri Nets.

Hongpeng Li received the B.S. degree in computer science and technology from University of Electronic Science and Technology of China in 2020, Zhongshan Institute. He is currently a master student in computer science and technology at the School of Computer Science and Technology, Guangdong University of Technology. His research interests include intelligent manufacturing scheduling and discrete event systems.

Cong Wang received the B.S. degree in manufacturing and automation and the M.S. degree in automotive engineering from Tsinghua University, in 2008 and 2010, respectively, the Ph.D. degree in mechanical engineering from University of California, USA in 2014. He was a Lecturer and Research Engineer at University of California, USA in 2015 before he joined New Jersey Institute of Technology, where he is currently an Associate Professor in electrical and computer engineering. His current research interests include nonlinear and data-driven control theories, robot physical intelligence, and crowdsourced human computation.

Yan Hou received the B.S. and M.S. degrees in computer science from Yangtze University, in 1999 and 2002, respectively, and the Ph.D. degree in industrial engineering from the Guangdong University of Technology in 2016. Since 2002, she has been with the School of Computer Science and Technology, Guangdong University of Technology where she is currently an Associate Professor. Her current research interests include production scheduling and optimization, information systems, and software engineering.